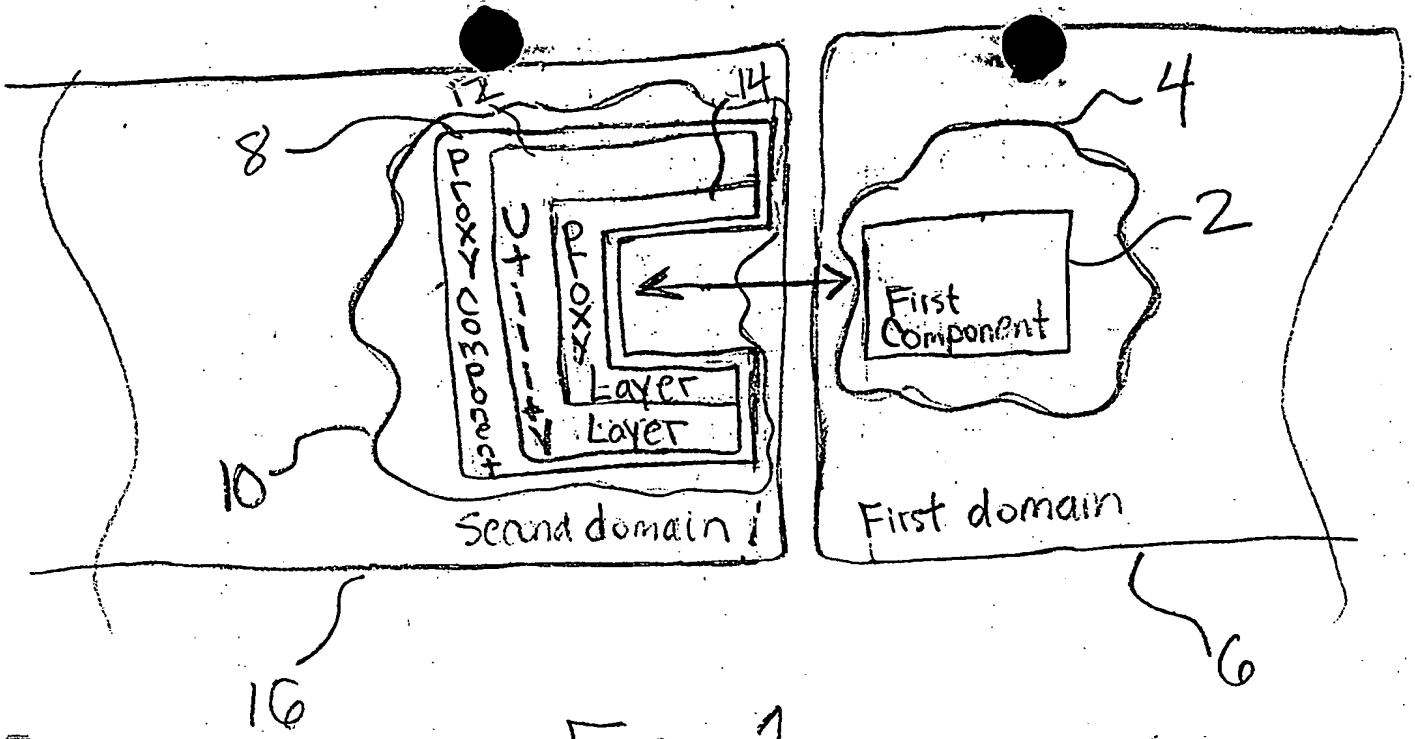

Fig 1

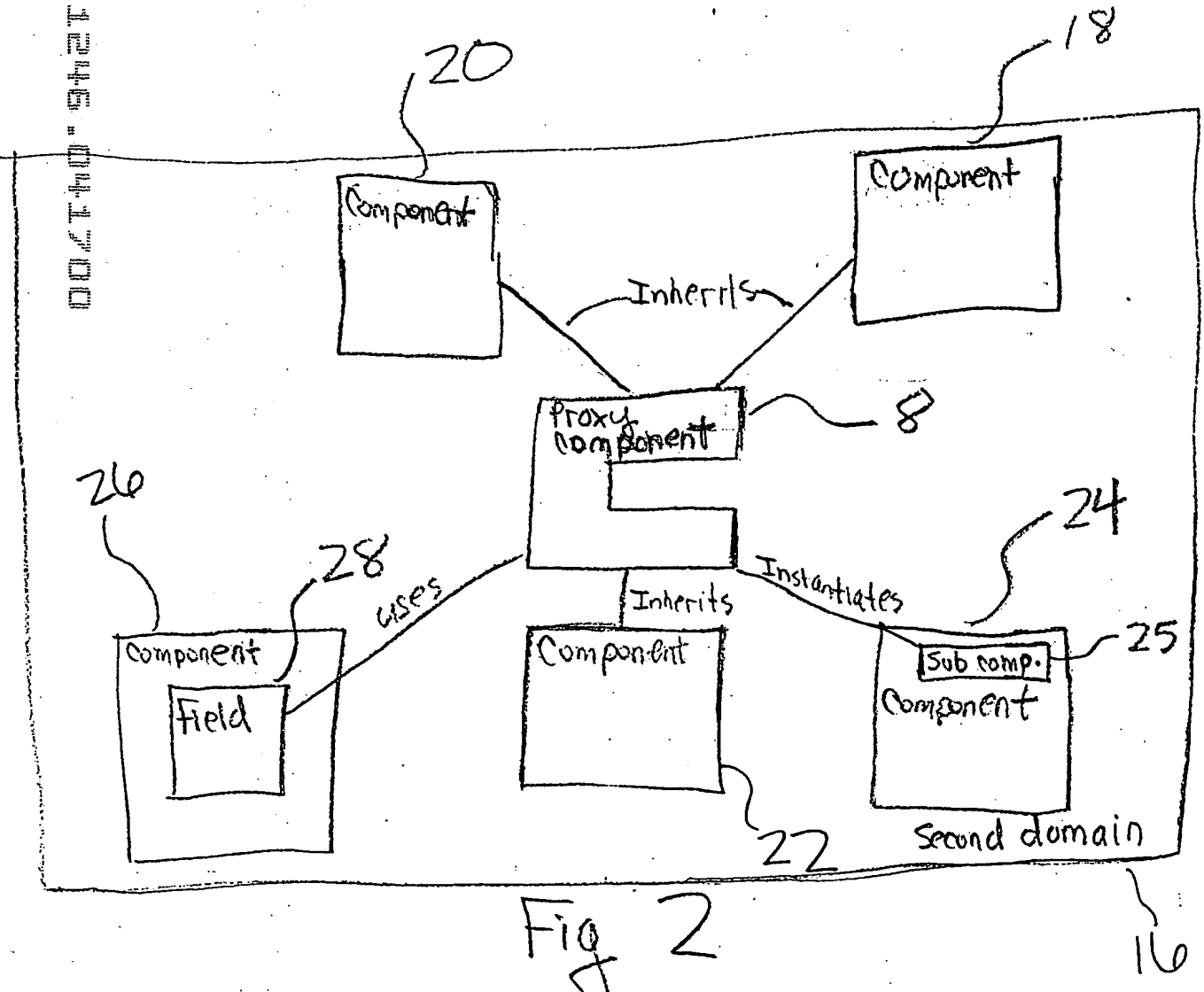

Fig 2

0055215.0047500

09551246 "0027A0

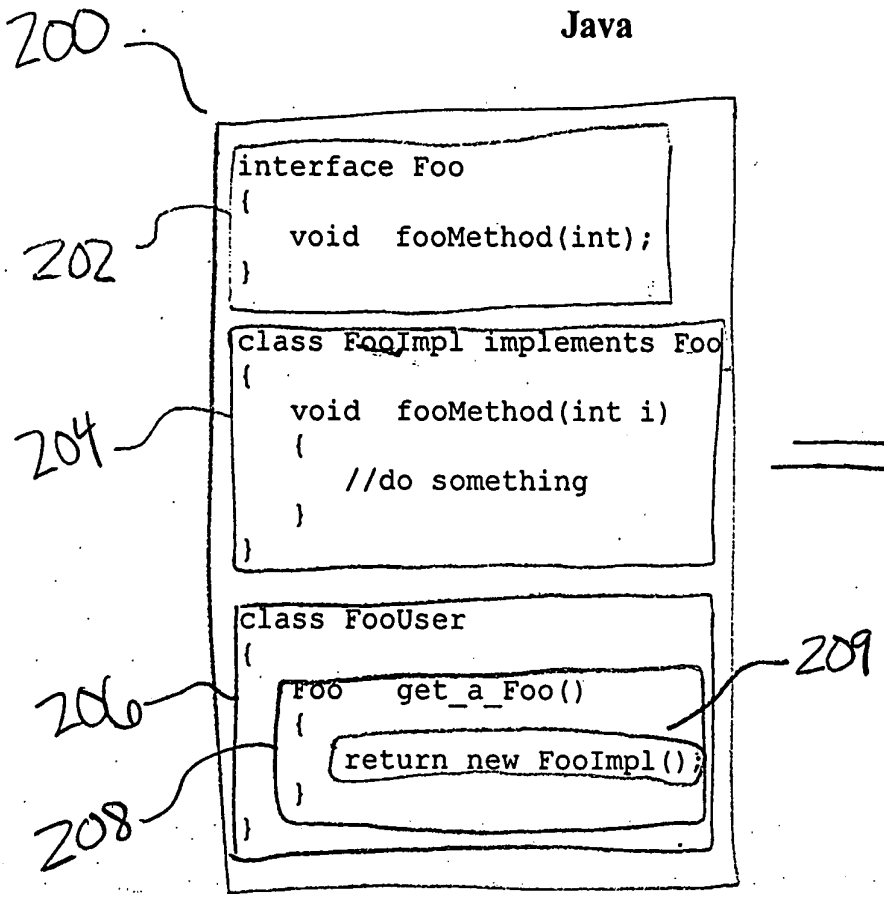

Fig. 3

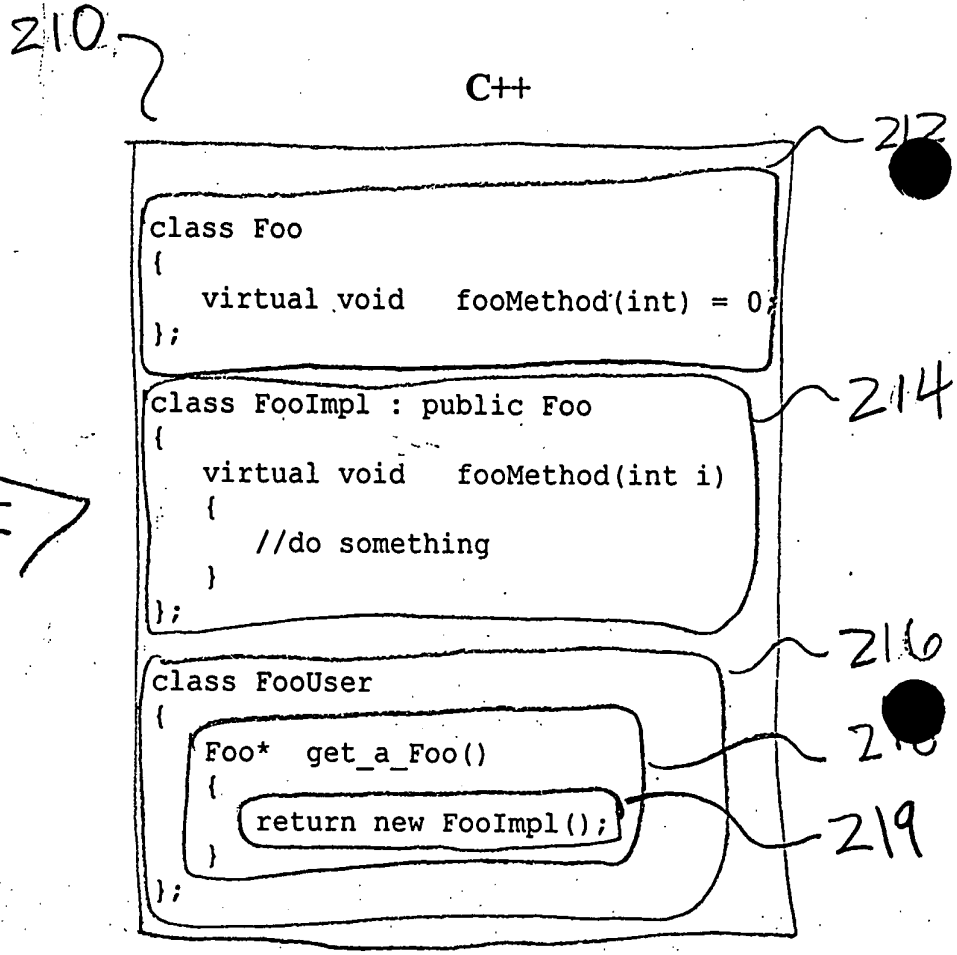

Fig. 4

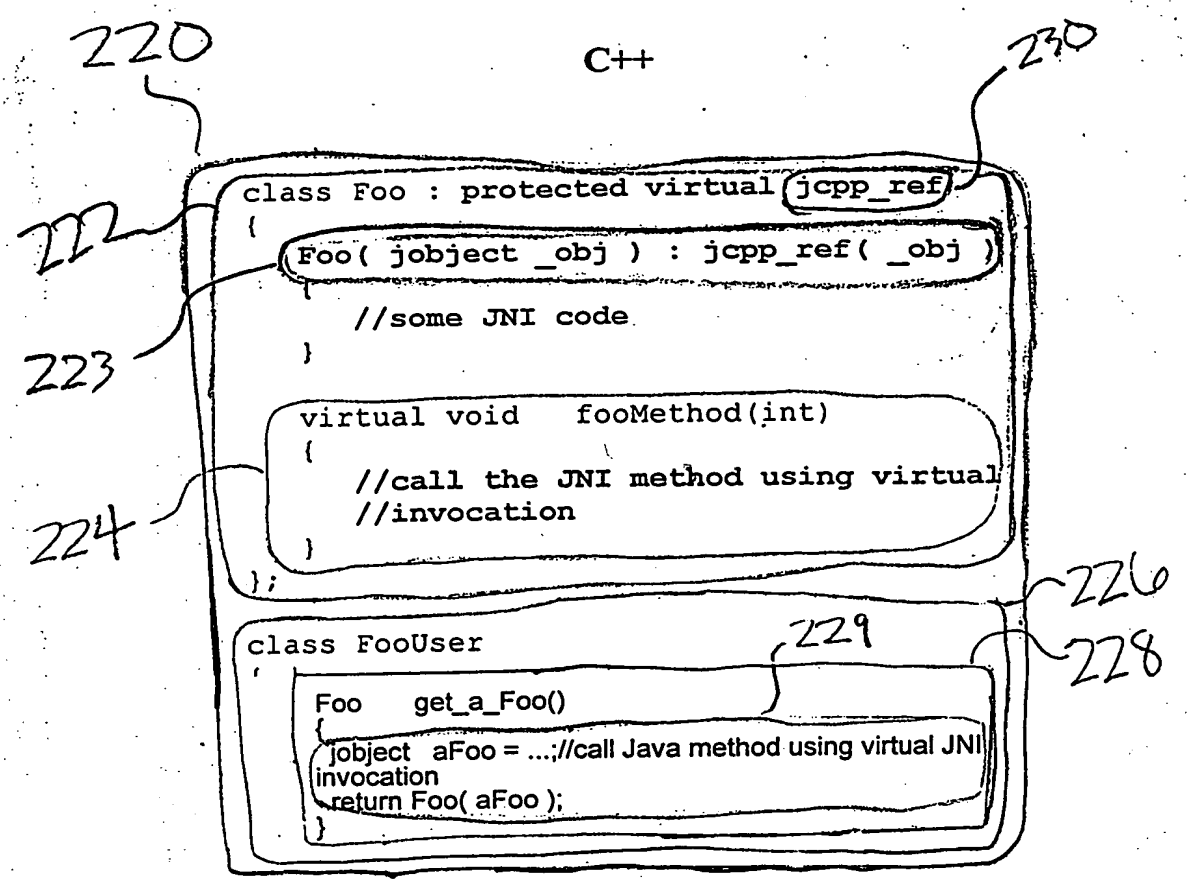
220
C++
230
222
223
224
226
228
229
class Foo : protected virtual jcpp_ref
{
Foo(jobject _obj) : jcpp_ref(_obj)
{
//some JNI code
}
virtual void fooMethod(int)
{
//call the JNI method using virtual
//invocation
}
};
class FooUser
Foo get_a_Foo()
{
jobject aFoo = ...;//call Java method using virtual JNI
invocation
return Foo(aFoo);
}

Fig. 5

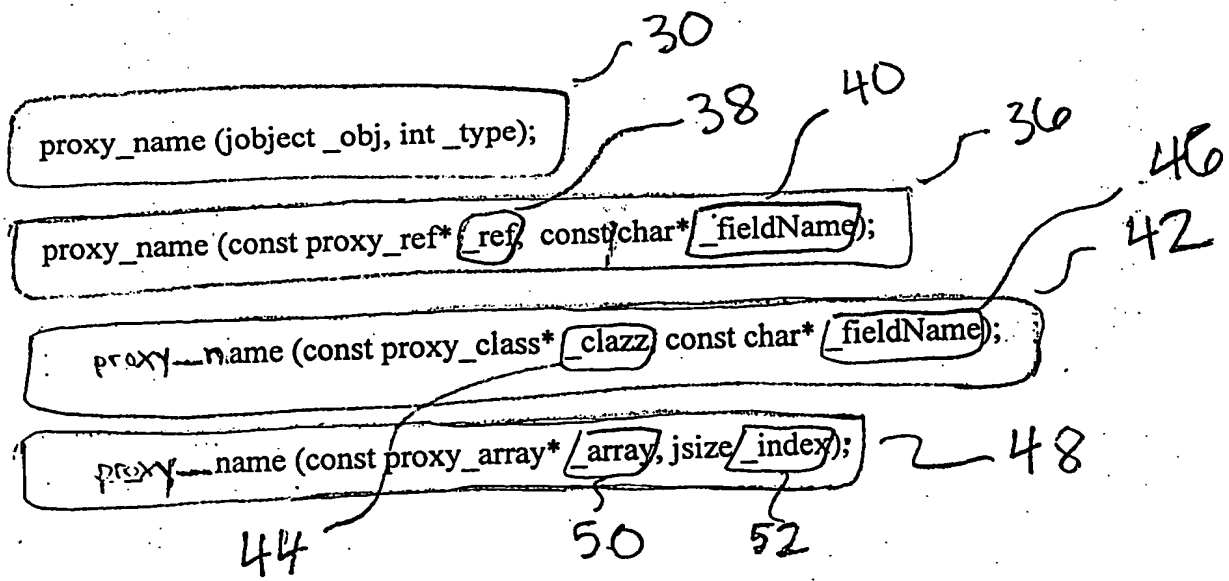
30
38
40
36
46
42
48
44
50
52
proxy_name (jobject _obj, int _type);
proxy_name (const proxy_ref* _ref, const char* _fieldName);
proxy_name (const proxy_class* _clazz, const char* _fieldName);
proxy_name (const proxy_array* _array, jsize _index);

Fig. 6

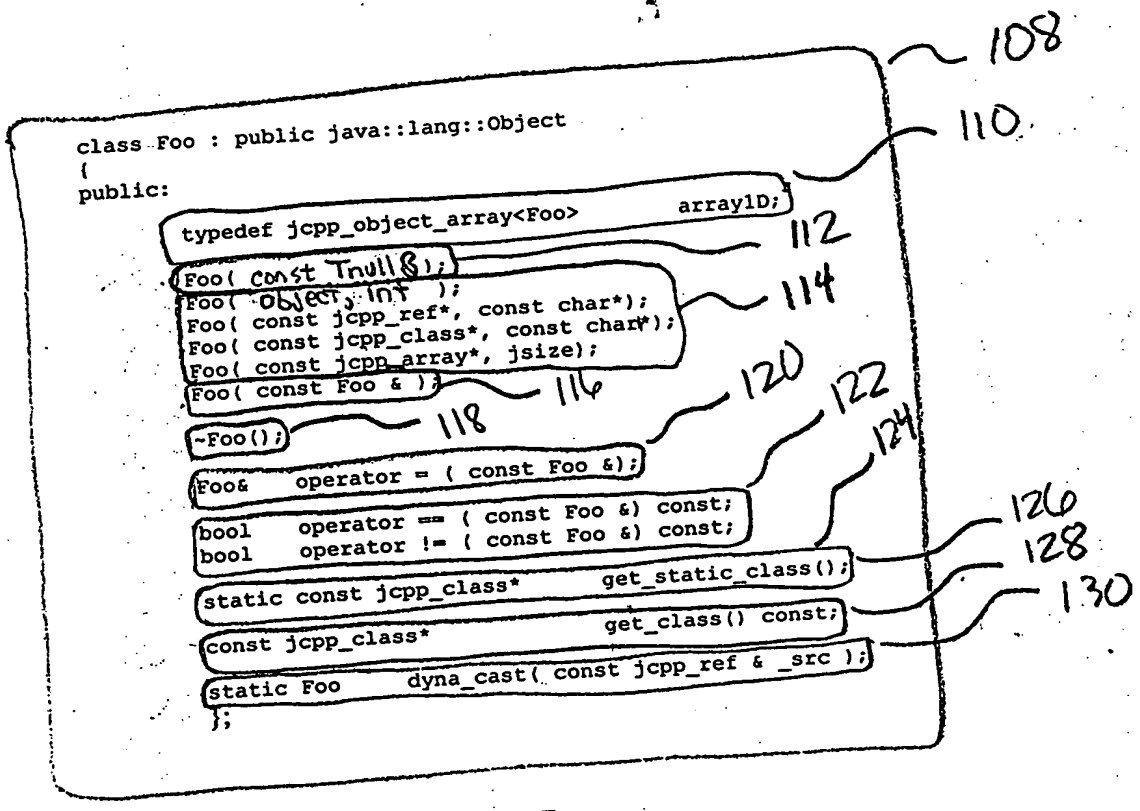
class Foo : public java::lang::Object
{
public:
typedef jcpp_object_array<Foo> arrayID;
Foo(const TnullS);
Foo(Object, int);
Foo(const jcpp_ref*, const char*);
Foo(const jcpp_class*, const char*);
Foo(const jcpp_array*, jsize);
Foo(const Foo &);
~Foo();
Foo& operator = (const Foo &);
bool operator == (const Foo &) const;
bool operator != (const Foo &) const;
static const jcpp_class* get_static_class();
const jcpp_class* get_class() const;
static Foo dyna_cast(const jcpp_ref & _src);
};
108
110
112
114
116
118
120
122
124
126
128
130

Fig. 7

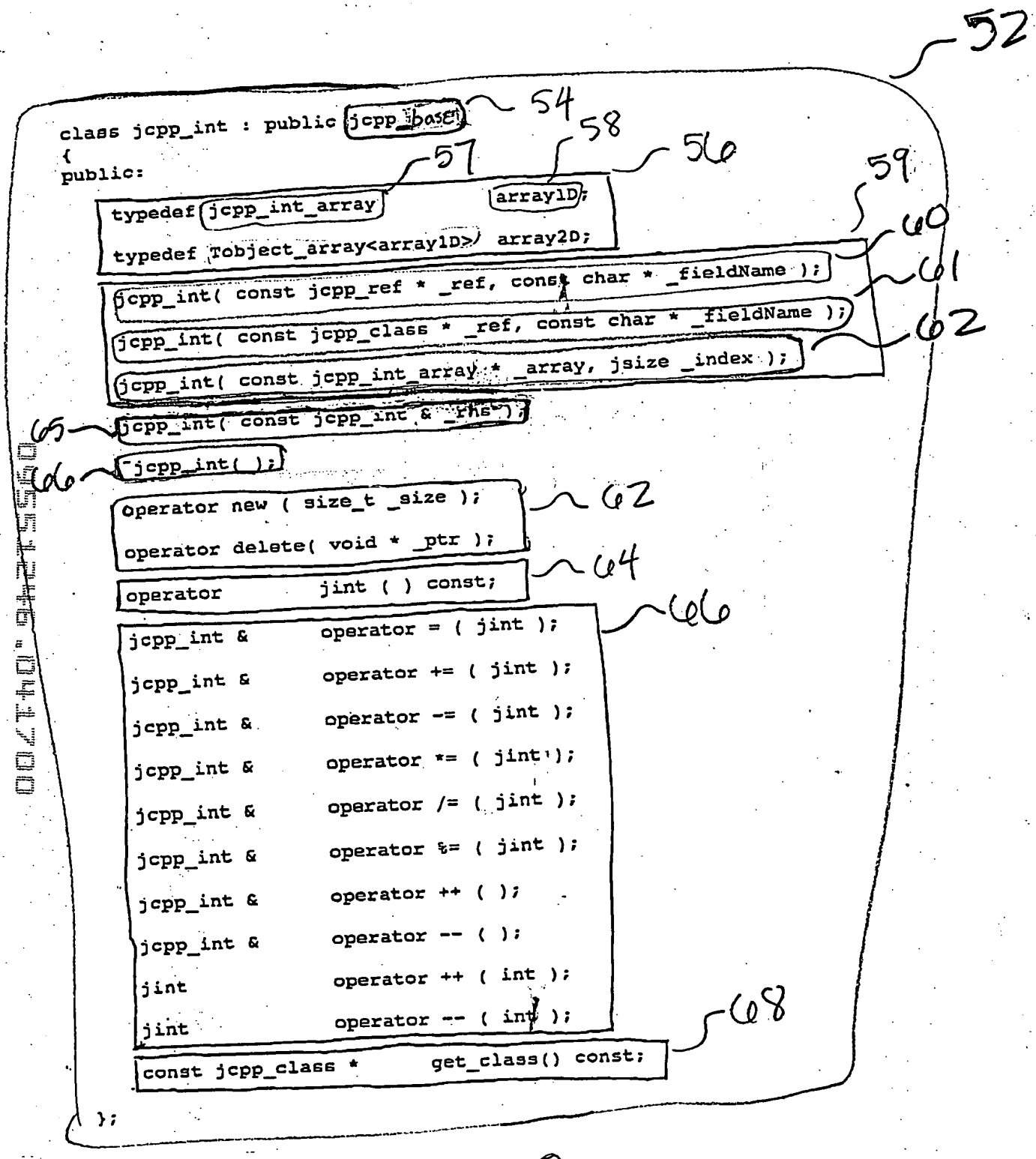
52
54
58
56
57
59
60
61
62
65
66
62
64
66
68
class jcpp_int : public jcpp_base
{
public:
typedef jcpp_int_array
array1D;
typedef Tobject_array<array1D> array2D;
jcpp_int(const jcpp_ref * _ref, const char * _fieldName);
jcpp_int(const jcpp_class * _ref, const char * _fieldName);
jcpp_int(const jcpp_int_array * _array, jsize _index);
jcpp_int(const jcpp_int & _rhs);
~jcpp_int( );
operator new (size_t _size);
operator delete(void * _ptr);
operator jint ( ) const;
jcpp_int & operator = (jint);
jcpp_int & operator += (jint);
jcpp_int & operator -= (jint);
jcpp_int & operator *= (jint);
jcpp_int & operator /= (jint);
jcpp_int & operator %= (jint);
jcpp_int & operator ++ ( );
jcpp_int & operator -- ( );
jint operator ++ (int);
jint operator -- (int);
const jcpp_class * get_class() const;
};

Fig. 8

09551246 "0427002

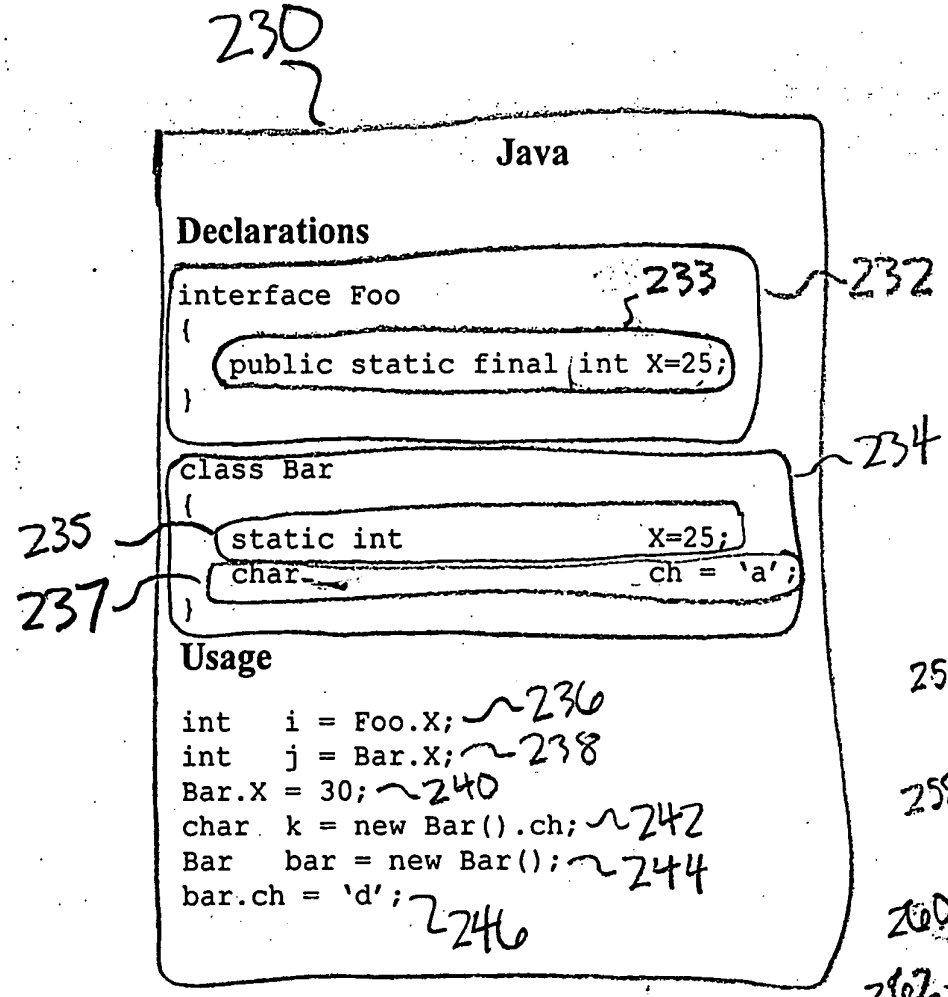

Fig. 9

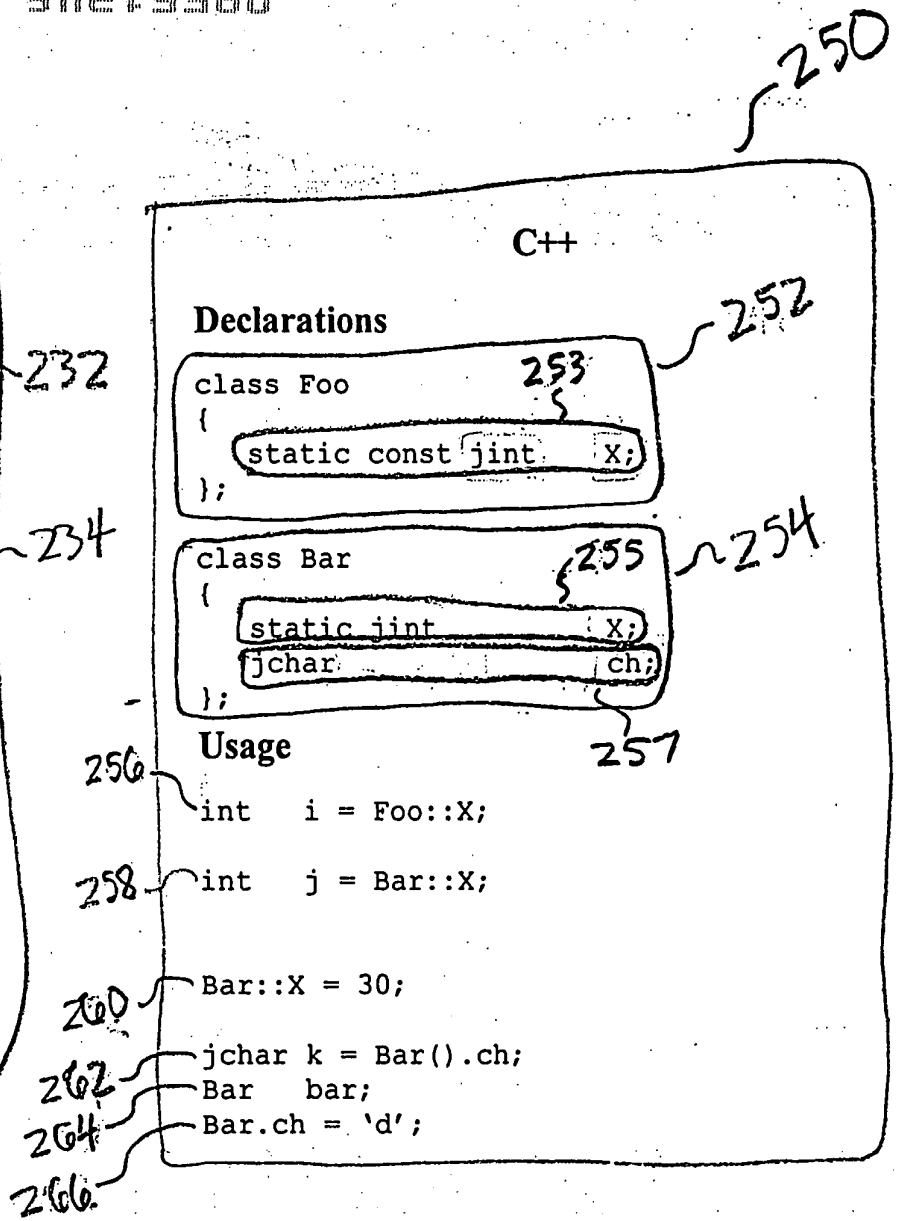

Fig. 10

09551246 "0h2700

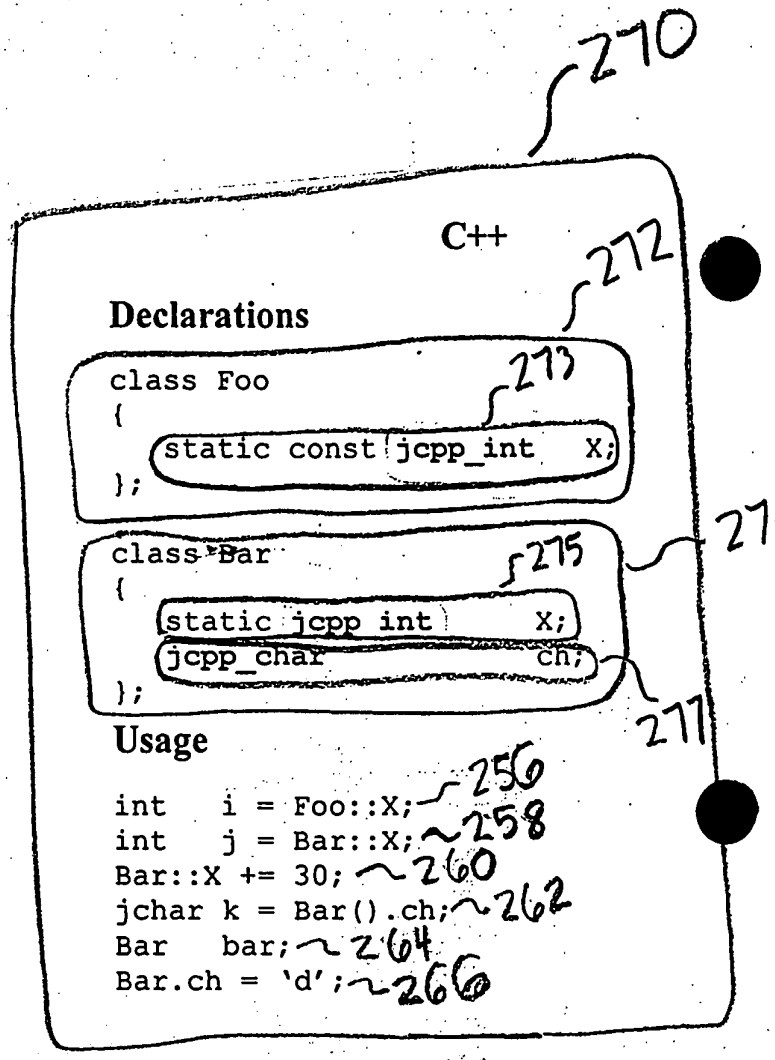

Fig. 11

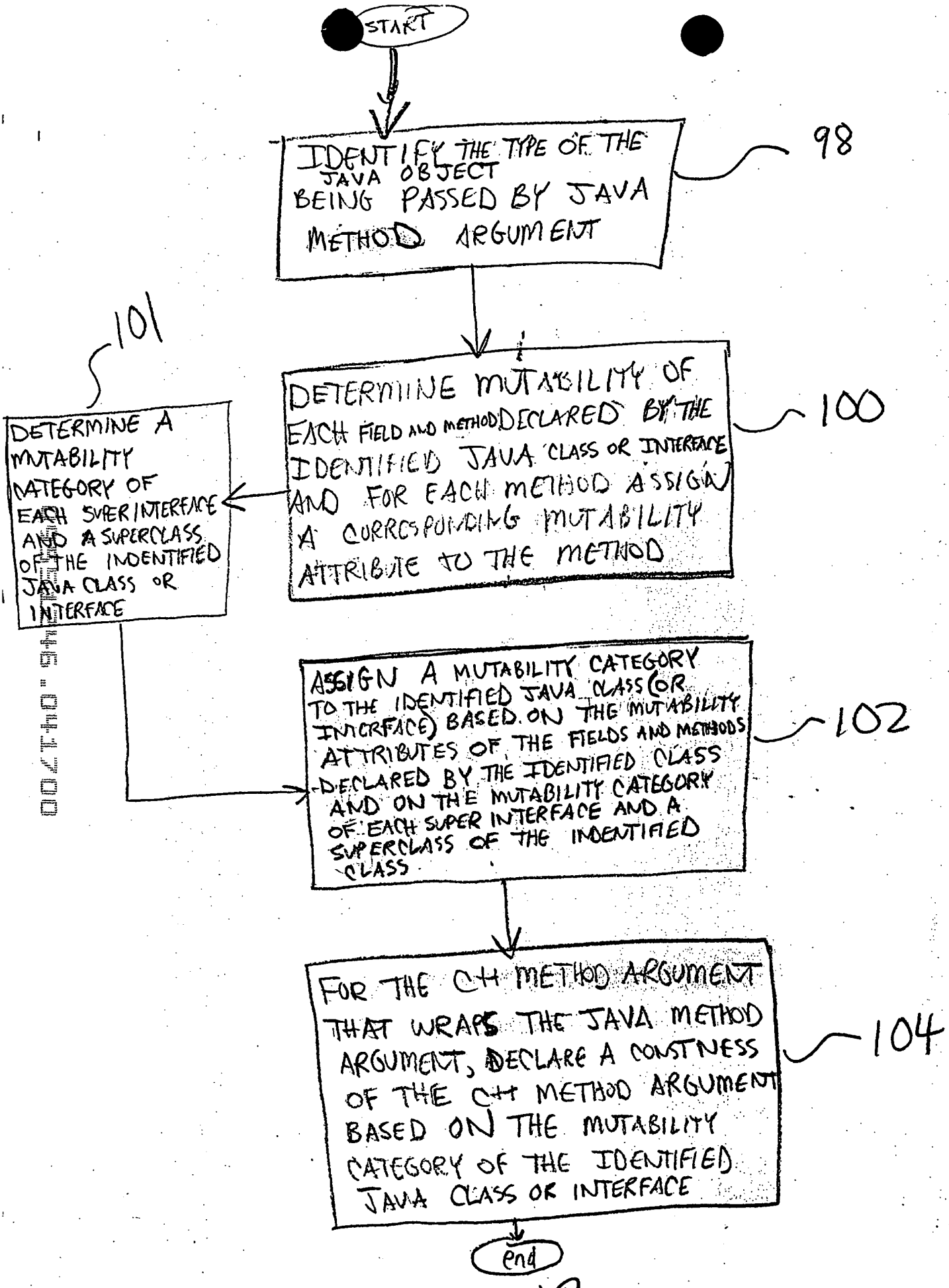
START
IDENTIFY THE TYPE OF THE JAVA OBJECT BEING PASSED BY JAVA METHOD ARGUMENT
98
DETERMINE MUTABILITY OF EACH FIELD AND METHOD DECLARED BY THE IDENTIFIED JAVA CLASS OR INTERFACE AND FOR EACH METHOD ASSIGN A CORRESPONDING MUTABILITY ATTRIBUTE TO THE METHOD
100
101
DETERMINE A MUTABILITY CATEGORY OF EACH SUPERINTERFACE AND A SUPERCLASS OF THE INDENTIFIED JAVA CLASS OR INTERFACE
ASSIGN A MUTABILITY CATEGORY TO THE IDENTIFIED JAVA CLASS (OR INTERFACE) BASED ON THE MUTABILITY ATTRIBUTES OF THE FIELDS AND METHODS DECLARED BY THE IDENTIFIED CLASS AND ON THE MUTABILITY CATEGORY OF EACH SUPER INTERFACE AND A SUPERCLASS OF THE INDENTIFIED CLASS
102
FOR THE C++ METHOD ARGUMENT THAT WRAPS THE JAVA METHOD ARGUMENT, DECLARE A CONSTNESS OF THE C++ METHOD ARGUMENT BASED ON THE MUTABILITY CATEGORY OF THE IDENTIFIED JAVA CLASS OR INTERFACE
104
end

Fig 12.

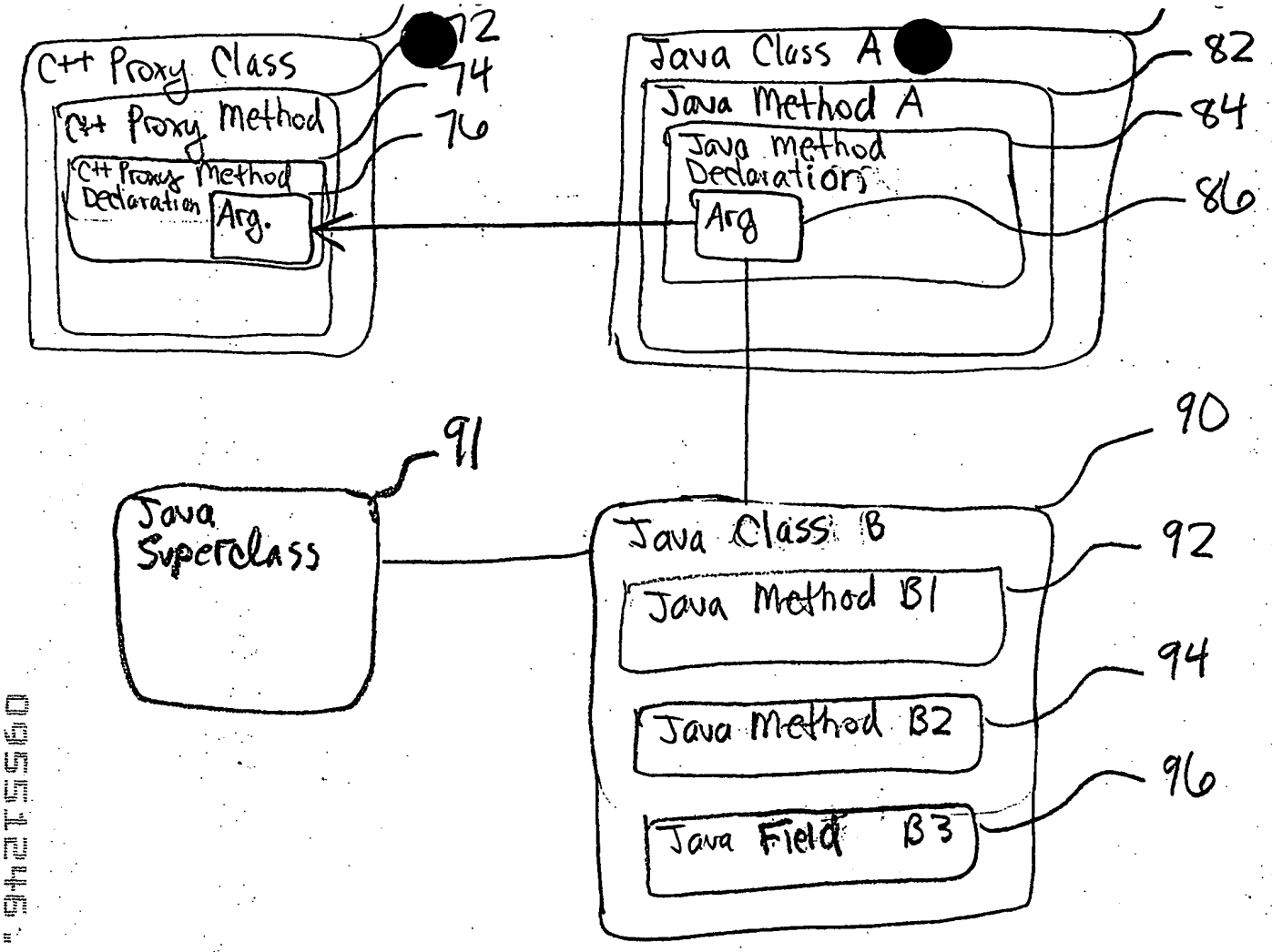

Fig 13

0055124 "04/7/00

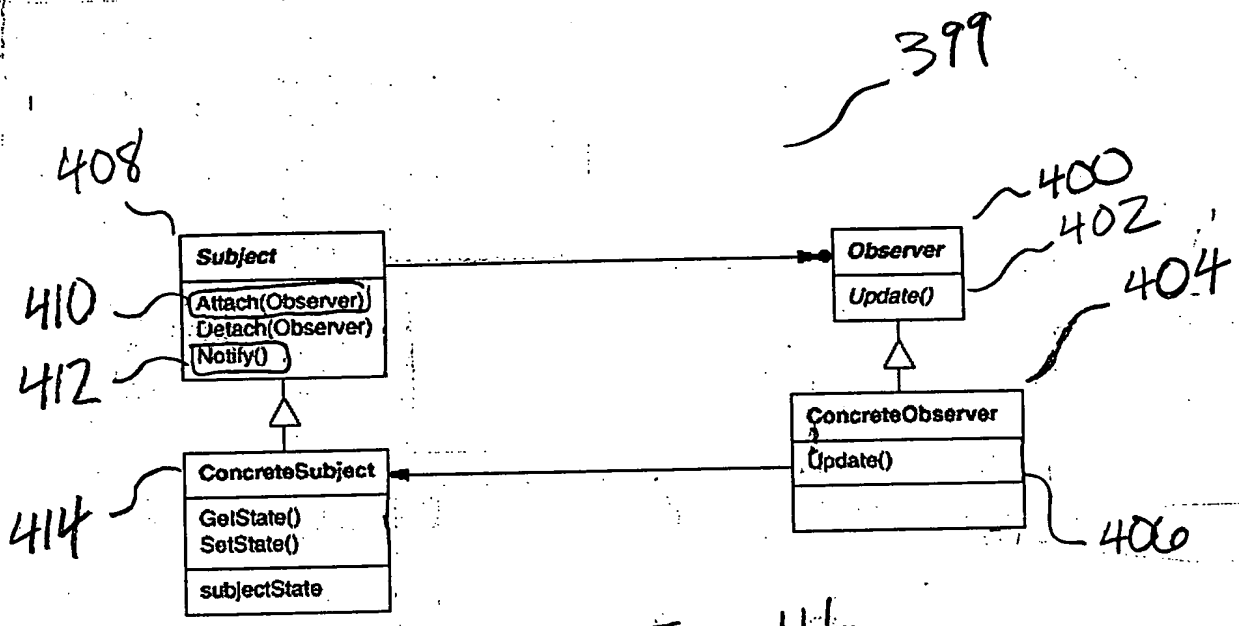
399
408
400
402
404
Subject
Attach(Observer)
Detach(Observer)
Notify()
410
412
Observer
Update()
ConcreteObserver
Update()
406
ConcreteSubject
GetState()
SetState()
subjectState
414

Fig 14

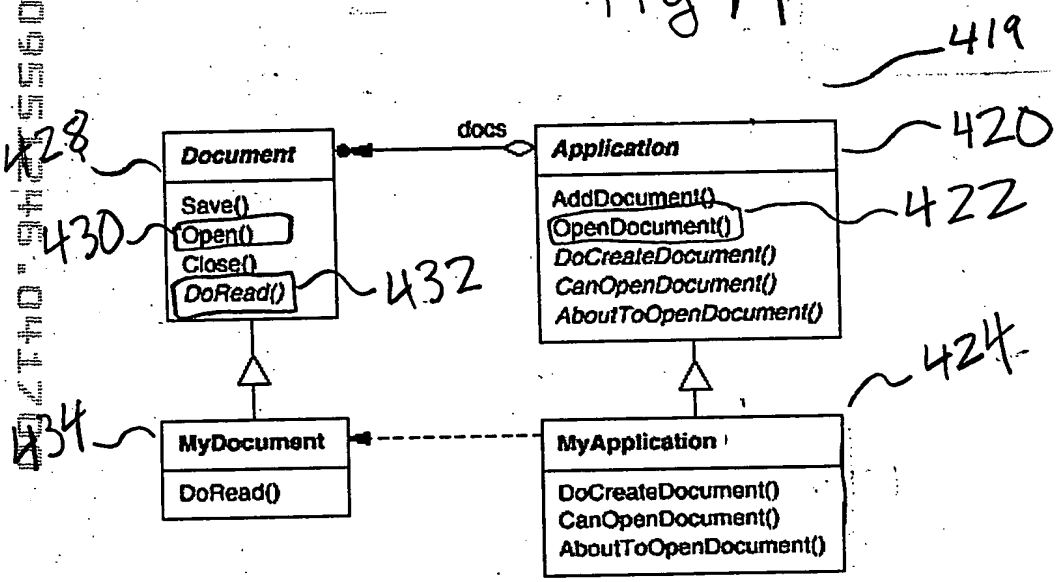
419
428
Document
Save()
Open()
Close()
DoRead()
430
432
docs
Application
AddDocument()
OpenDocument()
DoCreateDocument()
CanOpenDocument()
AboutToOpenDocument()
420
422
434
MyDocument
DoRead()
MyApplication
DoCreateDocument()
CanOpenDocument()
AboutToOpenDocument()
424

Fig 15

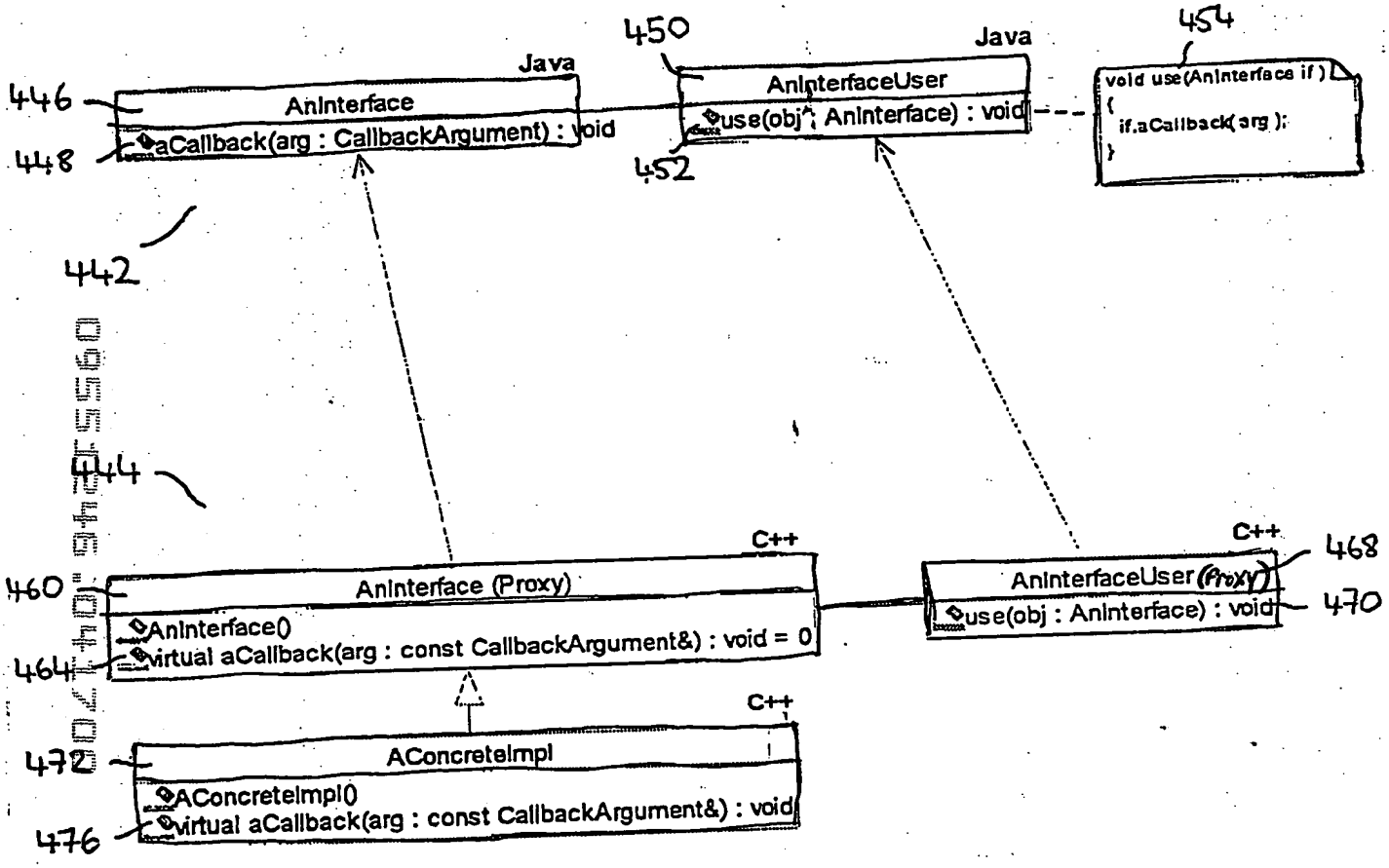
Java
AnInterface
aCallback(arg : CallbackArgument) : void
446
448
442
450
Java
AnInterfaceUser
use(obj : AnInterface) : void
452
454
void use(AnInterface if)
{
if.aCallback(arg);
}
444
C++
AnInterface (Proxy)
AnInterface()
virtual aCallback(arg : const CallbackArgument&) : void = 0
460
464
C++
AConcreteImpl
AConcreteImpl()
virtual aCallback(arg : const CallbackArgument&) : void
472
476
C++
AnInterfaceUser (Proxy)
use(obj : AnInterface) : void
468
470

Fig 16

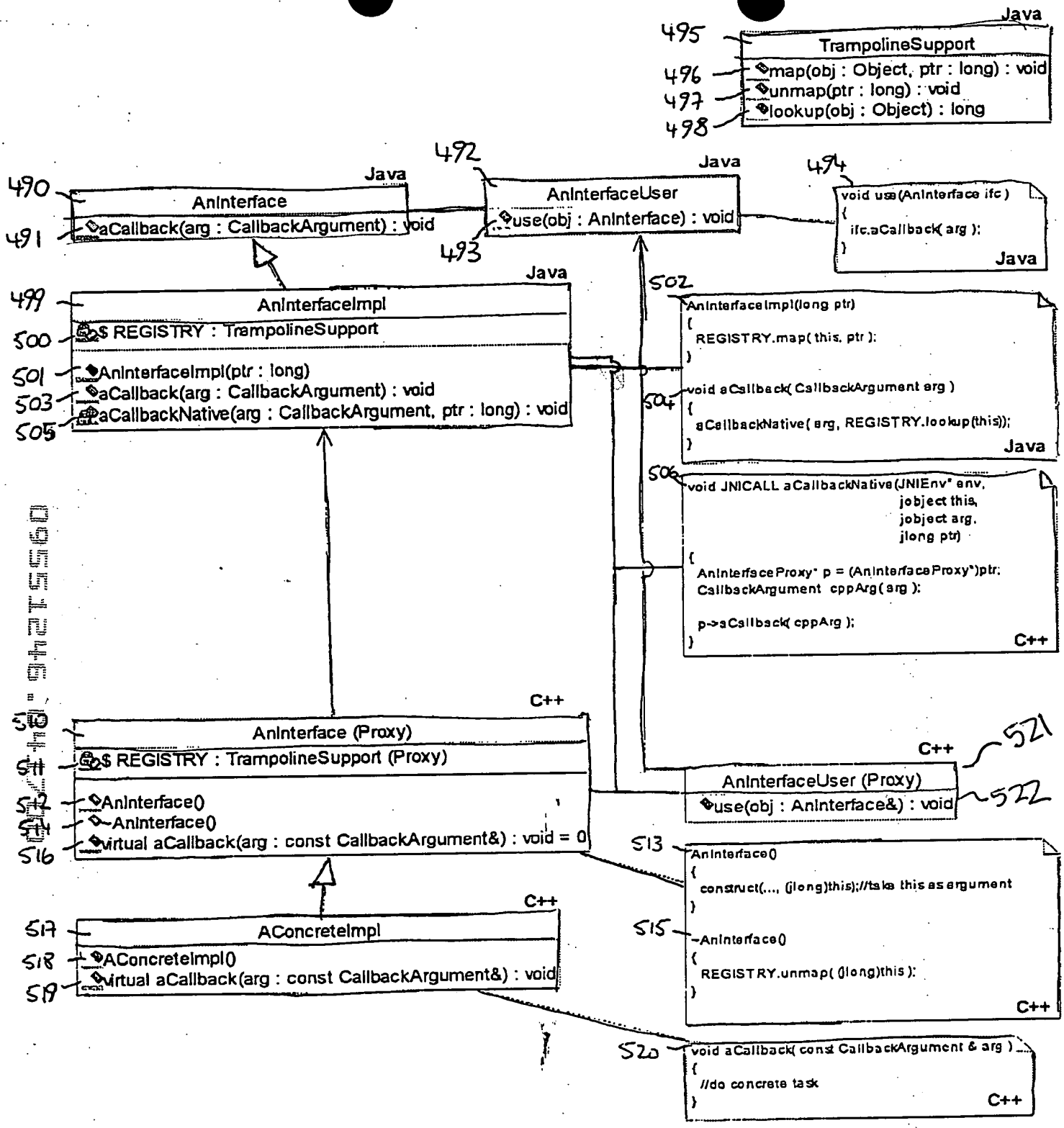
Java
TrampolineSupport
map(obj : Object, ptr : long) : void
unmap(ptr : long) : void
lookup(obj : Object) : long
AnInterface
aCallback(arg : CallbackArgument) : void
AnInterfaceUser
use(obj : AnInterface) : void
void use(AnInterface ifc)
{
ifc.aCallback(arg);
}
AnInterfaceImpl
$ REGISTRY : TrampolineSupport
AnInterfaceImpl(ptr : long)
aCallback(arg : CallbackArgument) : void
aCallbackNative(arg : CallbackArgument, ptr : long) : void
AnInterfaceImpl(long ptr)
{
REGISTRY.map(this, ptr);
}
void aCallback(CallbackArgument arg)
{
aCallbackNative(arg, REGISTRY.lookup(this));
}
void JNICALL aCallbackNative(JNIEnv* env,
jobject this,
jobject arg,
jlong ptr)
{
AnInterfaceProxy* p = (AnInterfaceProxy*)ptr;
CallbackArgument cppArg(arg);
p->aCallback(cppArg);
}
C++
AnInterface (Proxy)
$ REGISTRY : TrampolineSupport (Proxy)
AnInterface()
~AnInterface()
virtual aCallback(arg : const CallbackArgument&) : void = 0
AnInterfaceUser (Proxy)
use(obj : AnInterface&) : void
AnInterface()
{
construct(..., (jlong)this);//take this as argument
}
~AnInterface()
{
REGISTRY.unmap((jlong)this);
}
AConcreteImpl
AConcreteImpl()
virtual aCallback(arg : const CallbackArgument&) : void
void aCallback(const CallbackArgument & arg)
{
//do concrete task
}
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
510
511
512
513
514
515
516
517
518
519
520
521
522

Fig. 17

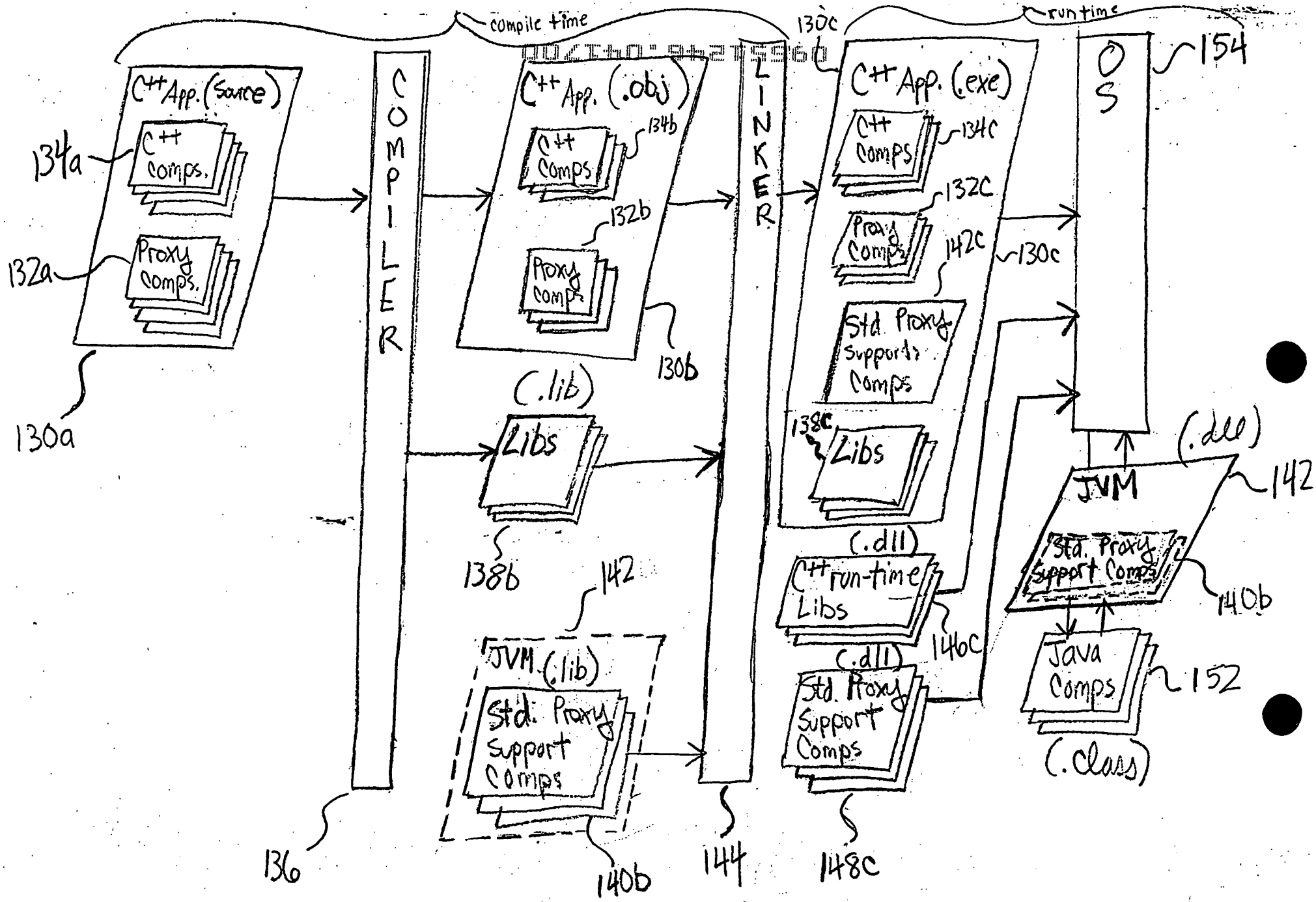
compile time
run time
C++ App. (source)
C++ Comps.
Proxy Comps.
134a
132a
130a
COMPILER
C++ App. (.obj)
C++ Comps
134b
Proxy Comps
132b
130b
(.lib)
Libs
138b
142
JVM (.lib)
Std. Proxy Support Comps
140b
136
144
LINKER
130c
C++ App. (.exe)
C++ Comps
134c
Proxy Comps
132c
142c
Std. Proxy Support Comps
138c
Libs
(.dll)
C++ run-time Libs
146c
(.dll)
Std. Proxy Support Comps
148c
OS
154
(.dll)
JVM
142
Std. Proxy Support Comps
140b
Java Comps
152
(.class)

Fig. 18

00Th0" 9h2T5560

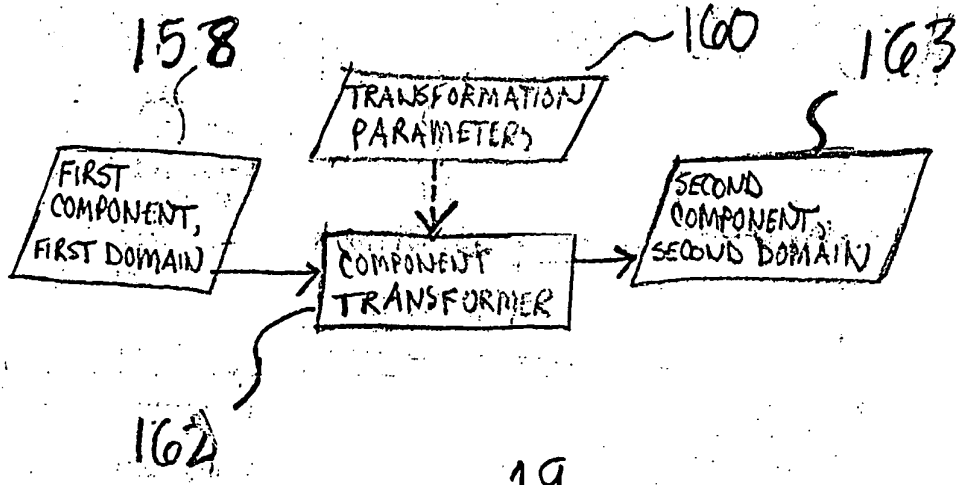

Fig. 19

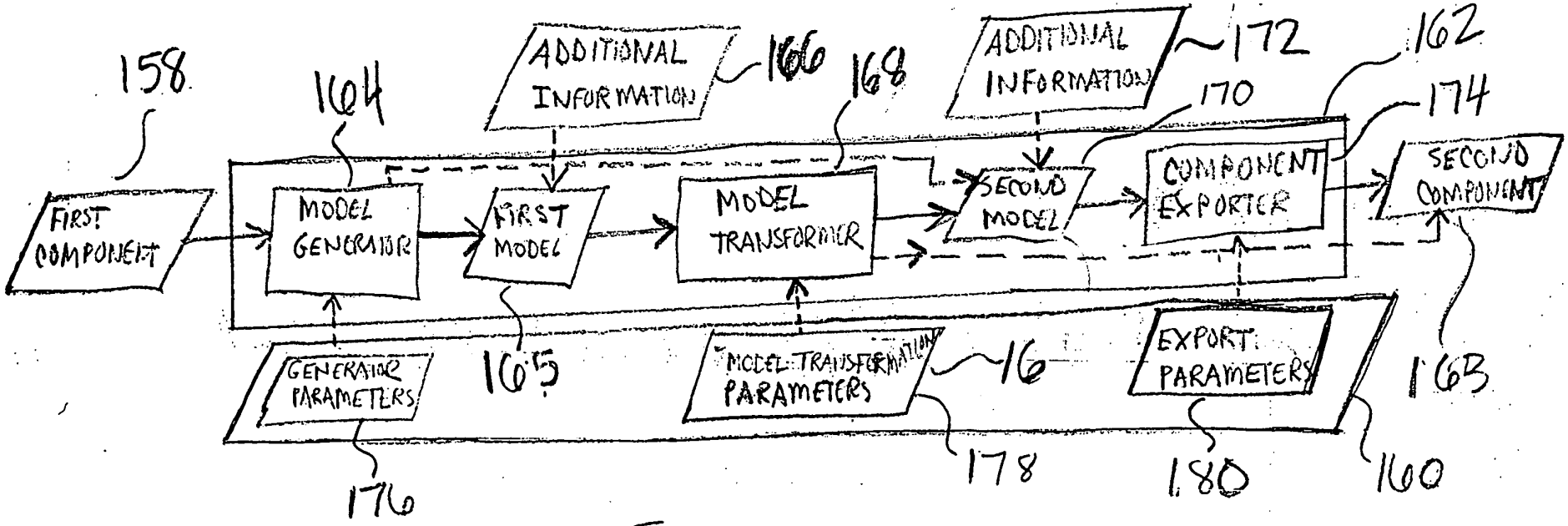

Fig. 23

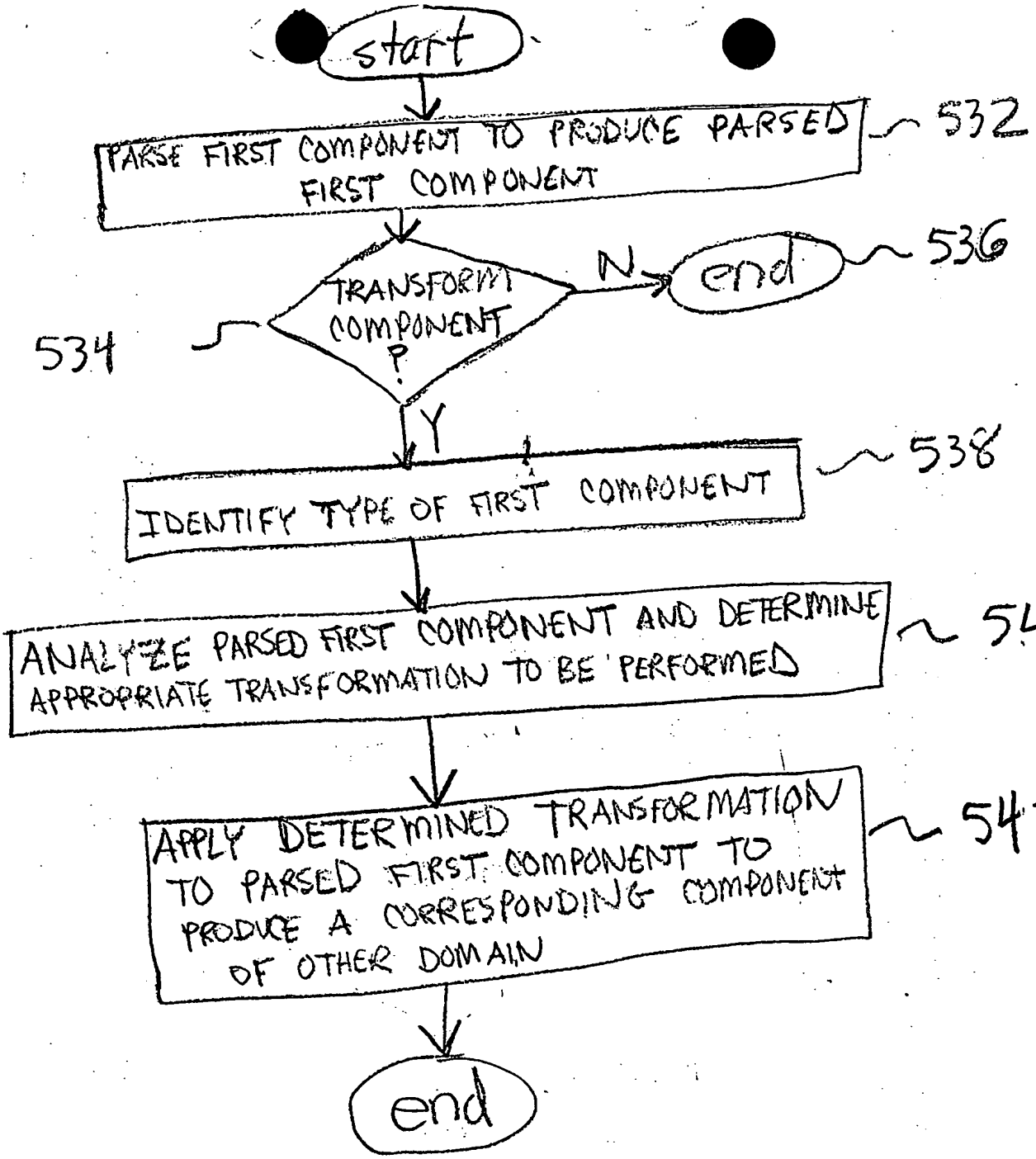
start
PARSE FIRST COMPONENT TO PRODUCE PARSED FIRST COMPONENT
532
TRANSFORM COMPONENT P.
534
N
end
536
Y
IDENTIFY TYPE OF FIRST COMPONENT
538
ANALYZE PARSED FIRST COMPONENT AND DETERMINE APPROPRIATE TRANSFORMATION TO BE PERFORMED
540
APPLY DETERMINED TRANSFORMATION TO PARSED FIRST COMPONENT TO PRODUCE A CORRESPONDING COMPONENT OF OTHER DOMAIN
542
end

Fig 20

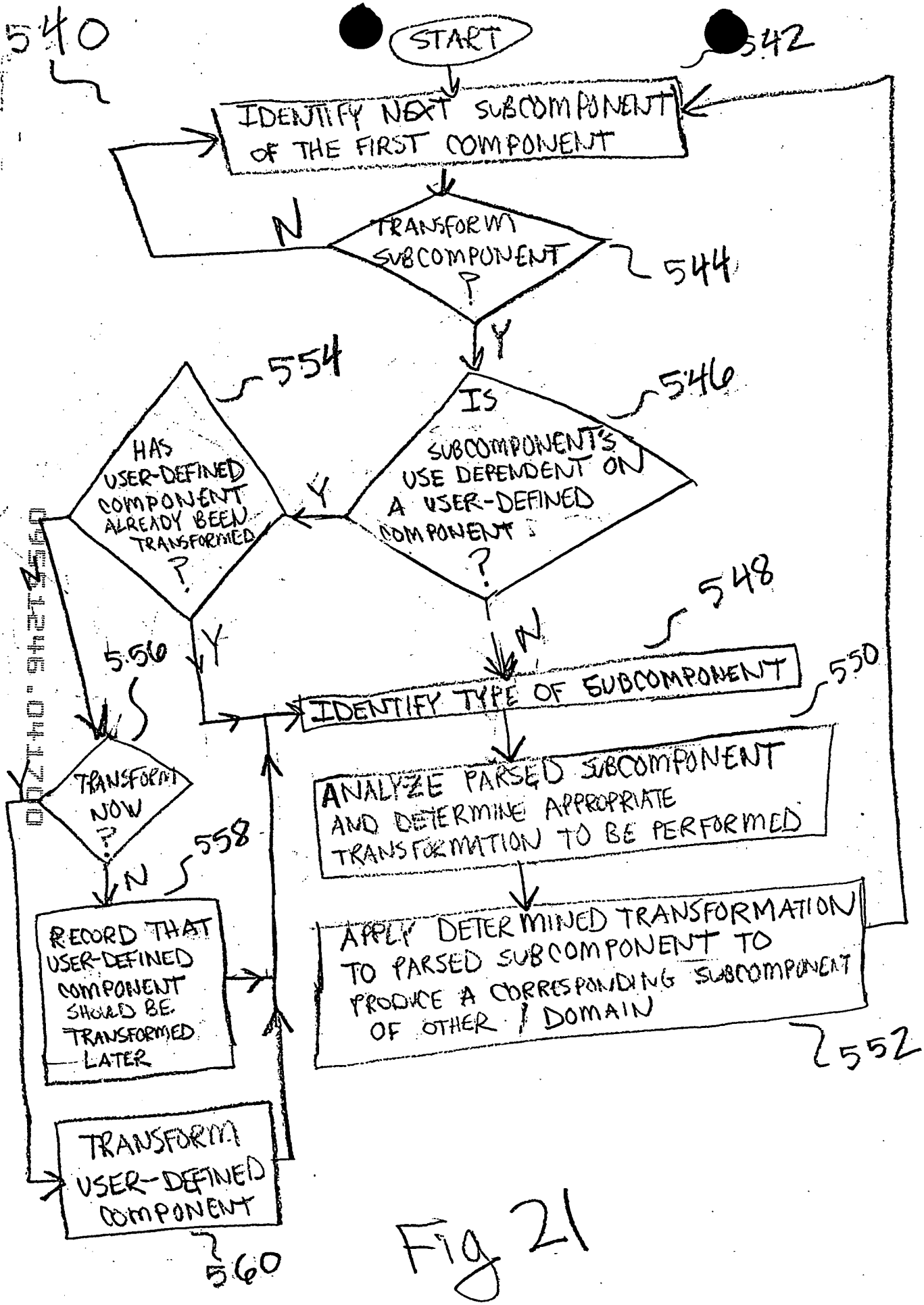
540
START
542
IDENTIFY NEXT SUBCOMPONENT OF THE FIRST COMPONENT
TRANSFORM SUBCOMPONENT ?
544
N
Y
554
HAS USER-DEFINED COMPONENT ALREADY BEEN TRANSFORMED ?
546
IS SUBCOMPONENT'S USE DEPENDENT ON A USER-DEFINED COMPONENT ?
Y
N
548
556
IDENTIFY TYPE OF SUBCOMPONENT
550
TRANSFORM NOW ?
Y
N
558
ANALYZE PARSED SUBCOMPONENT AND DETERMINE APPROPRIATE TRANSFORMATION TO BE PERFORMED
RECORD THAT USER-DEFINED COMPONENT SHOULD BE TRANSFORMED LATER
APPLY DETERMINED TRANSFORMATION TO PARSED SUBCOMPONENT TO PRODUCE A CORRESPONDING SUBCOMPONENT OF OTHER DOMAIN
552
TRANSFORM USER-DEFINED COMPONENT
560

Fig 21

0054705.04245.0950

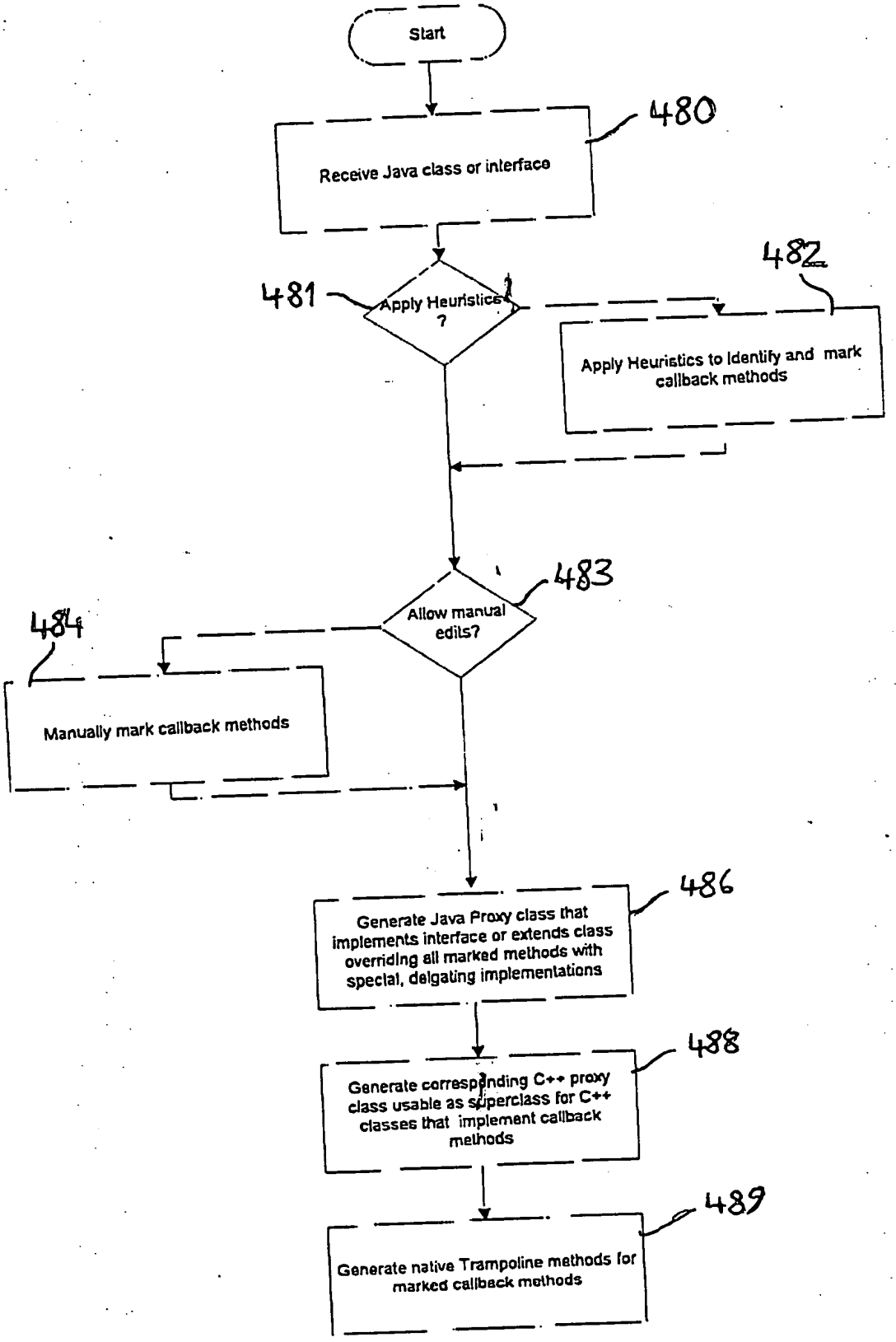

Fig. 22

09551246-041700

300a

```
public class Counter implements java.io.Serializable
{
    public static final int    UP = 1;          304a
    public static final int    DOWN = 2;        306a

    private int    max;                         308a
    private int    direction;                   310a

    //creates a new UP-counter with the given maximum
    public Counter( int _max )
    {
        this( _max, UP );
    }

    //creates a new counter with given maximum and direction
    public Counter( int _max, int _direction )
    {
        max = _max;
        direction = _direction;
    }

    //counts in the direction specified and outputs the numbers
    public void    count()
    {
        if( direction == UP )
            for( int I=0; I<max; I++ )
                System.out.println( " " + I );
        else if( direction == DOWN )
            for( int I=max-1; I>=0; I-- )
                System.out.println( " " + I );
    }

    //returns true if this instance is an UP counter
    public boolean     isUpCounter()
    {
        return ( direction == UP );
    }

    //returns the maximum of the counter
    public final int   getMax()
    {
        return max;
    }

    //creates a counter with the same maximum as this counter, but reverse direction
    public Counter     getReverseCounter()
    {
        return new Counter( max, direction == UP ? DOWN : UP );
    }
}
```

312a

314a

316a

318a

320a

322a

Fig. 24

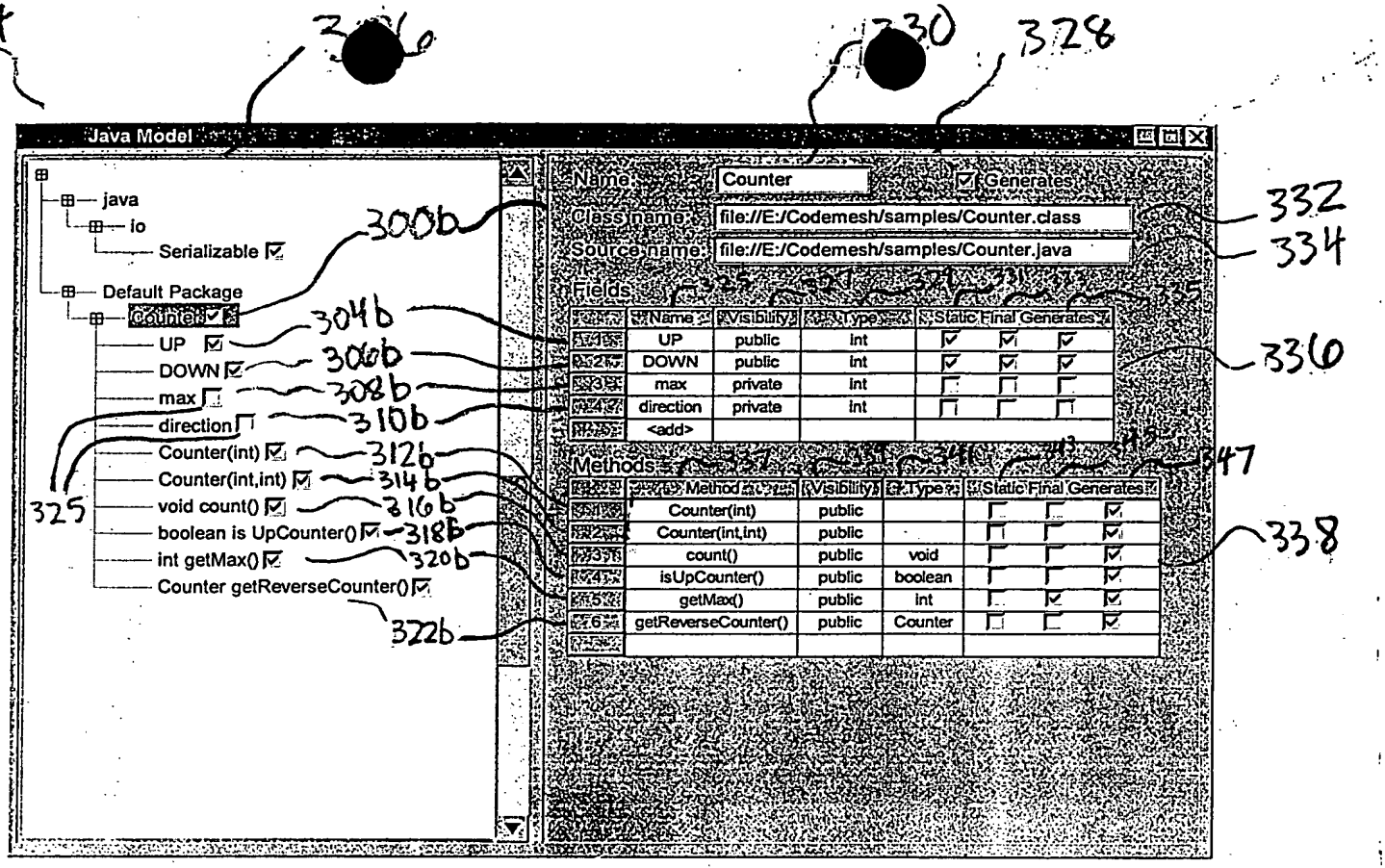
Java Model
java
io
Serializable
Default Package
Counter
UP
DOWN
max
direction
Counter(int)
Counter(int,int)
void count()
boolean is UpCounter()
int getMax()
Counter getReverseCounter()
Name: Counter
Generates
Class name: file://E:/Codemesh/samples/Counter.class
Source name: file://E:/Codemesh/samples/Counter.java
Fields
Name | Visibility | Type | Static | Final | Generates
UP | public | int
DOWN | public | int
max | private | int
direction | private | int
<add>
Methods
Method | Visibility | Type | Static | Final | Generates
Counter(int) | public
Counter(int,int) | public
count() | public | void
isUpCounter() | public | boolean
getMax() | public | int
getReverseCounter() | public | Counter
324
326
328
330
332
334
336
338
340
342
344
346
347
348
349
350
351
352
353
354
356
300b
304b
306b
308b
310b
312b
314b
316b
318b
320b
322b
325

Fig. 25

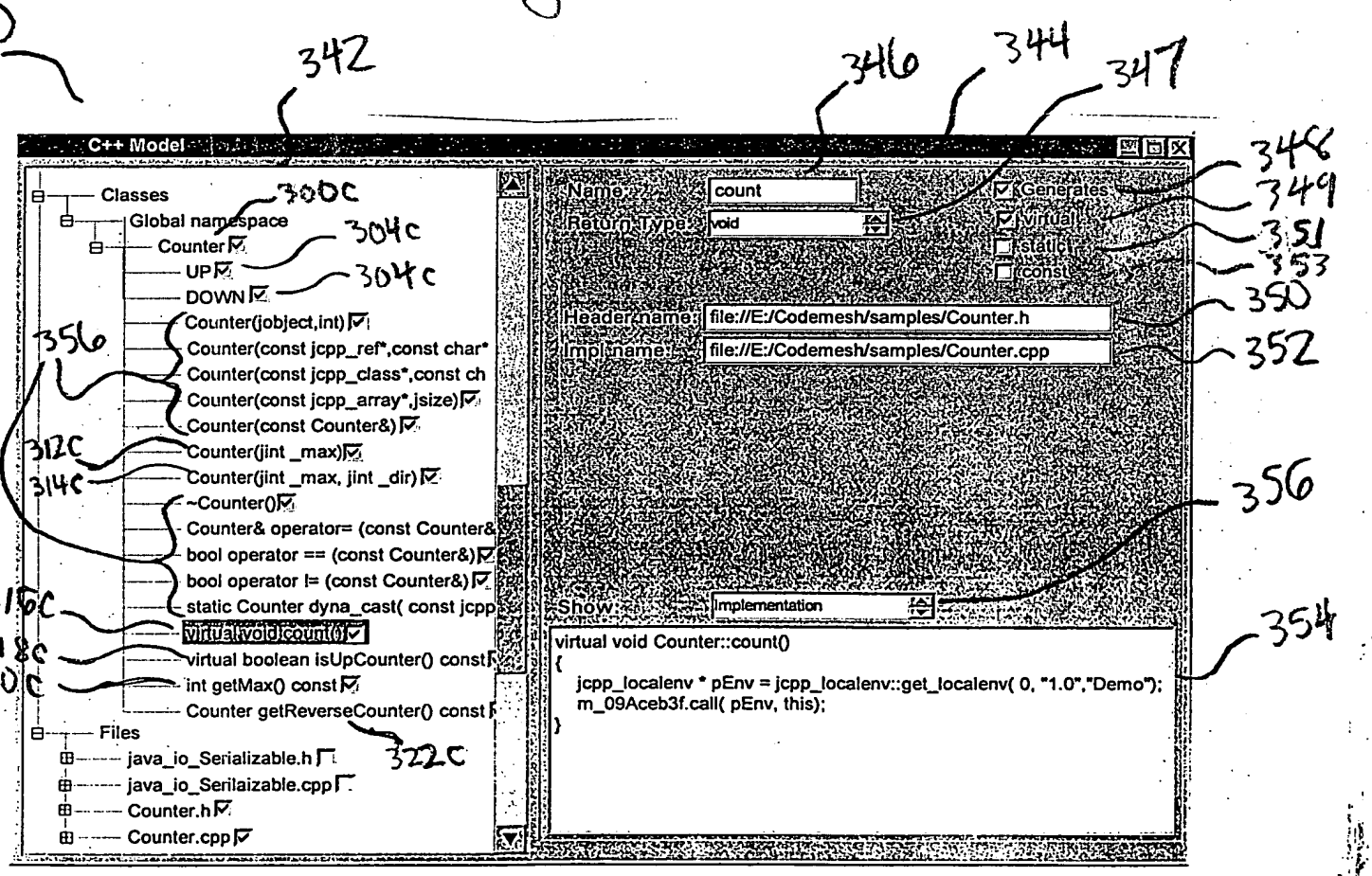
C++ Model
Classes
Global namespace
Counter
UP
DOWN
Counter(jobject,int)
Counter(const jcpp_ref*,const char*
Counter(const jcpp_class*,const ch
Counter(const jcpp_array*,jsize)
Counter(const Counter&)
Counter(jint _max)
Counter(jint _max, jint _dir)
~Counter()
Counter& operator= (const Counter&
bool operator == (const Counter&)
bool operator != (const Counter&)
static Counter dyna_cast(const jcpp
virtual void count()
virtual boolean isUpCounter() const
int getMax() const
Counter getReverseCounter() const
Files
java_io_Serializable.h
java_io_Serilaizable.cpp
Counter.h
Counter.cpp
Name: count
Generates
Return Type: void
virtual
static
const
Header name: file://E:/Codemesh/samples/Counter.h
Impl name: file://E:/Codemesh/samples/Counter.cpp
Show: implementation
virtual void Counter::count()
{
jcpp_localenv * pEnv = jcpp_localenv::get_localenv(0, "1.0","Demo");
m_09Aceb3f.call(pEnv, this);
}
340
342
344
346
347
348
349
350
351
352
353
354
356
300c
304c
306c
312c
314c
316c
318c
320c
322c

Fig. 26

0027h0" 9h2T5560

Bottom-Up Port By Proxy (Initial)

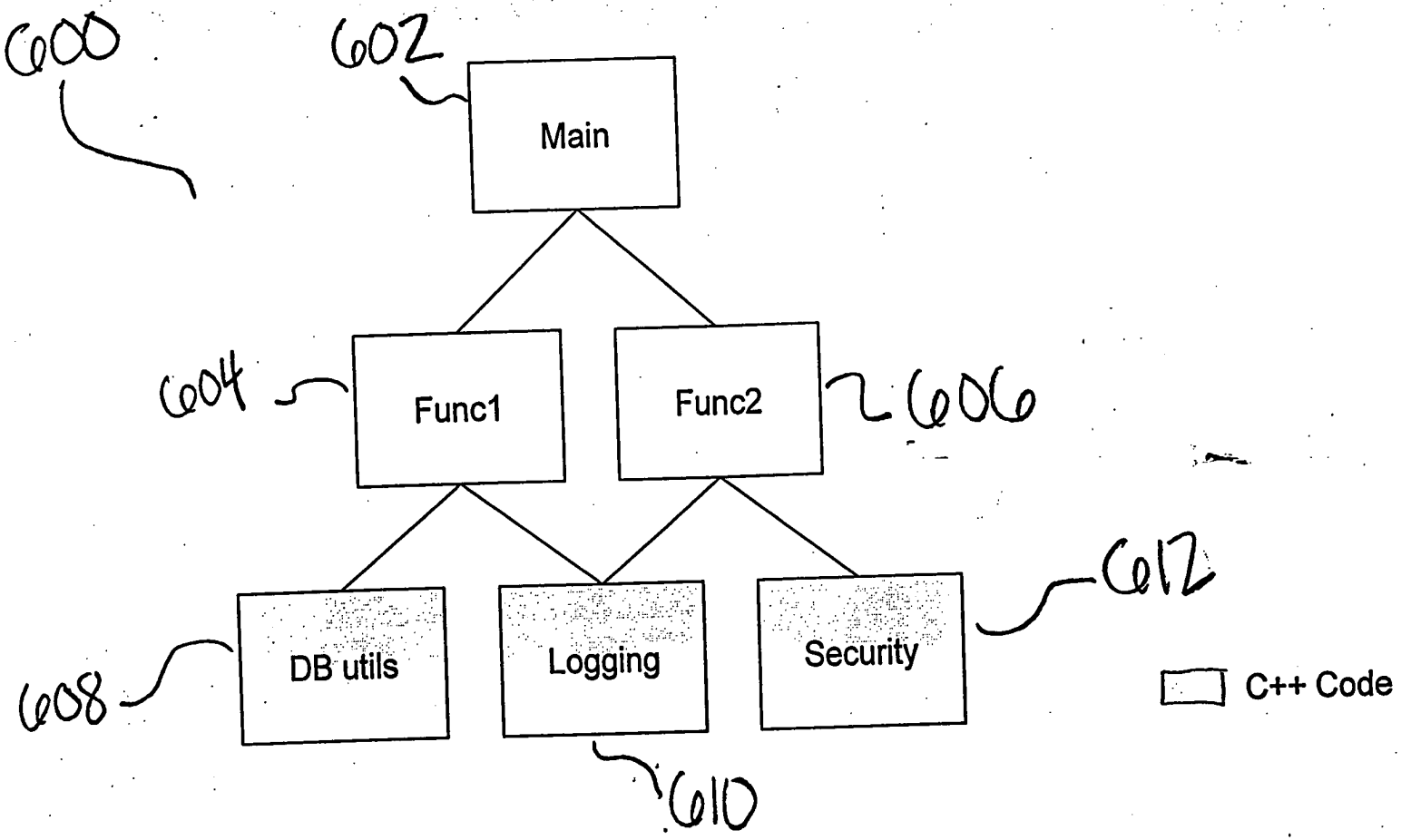

Fig. 27a

09551246 "0h2T5960

Bottom-Up Port By Proxy (1st Step)

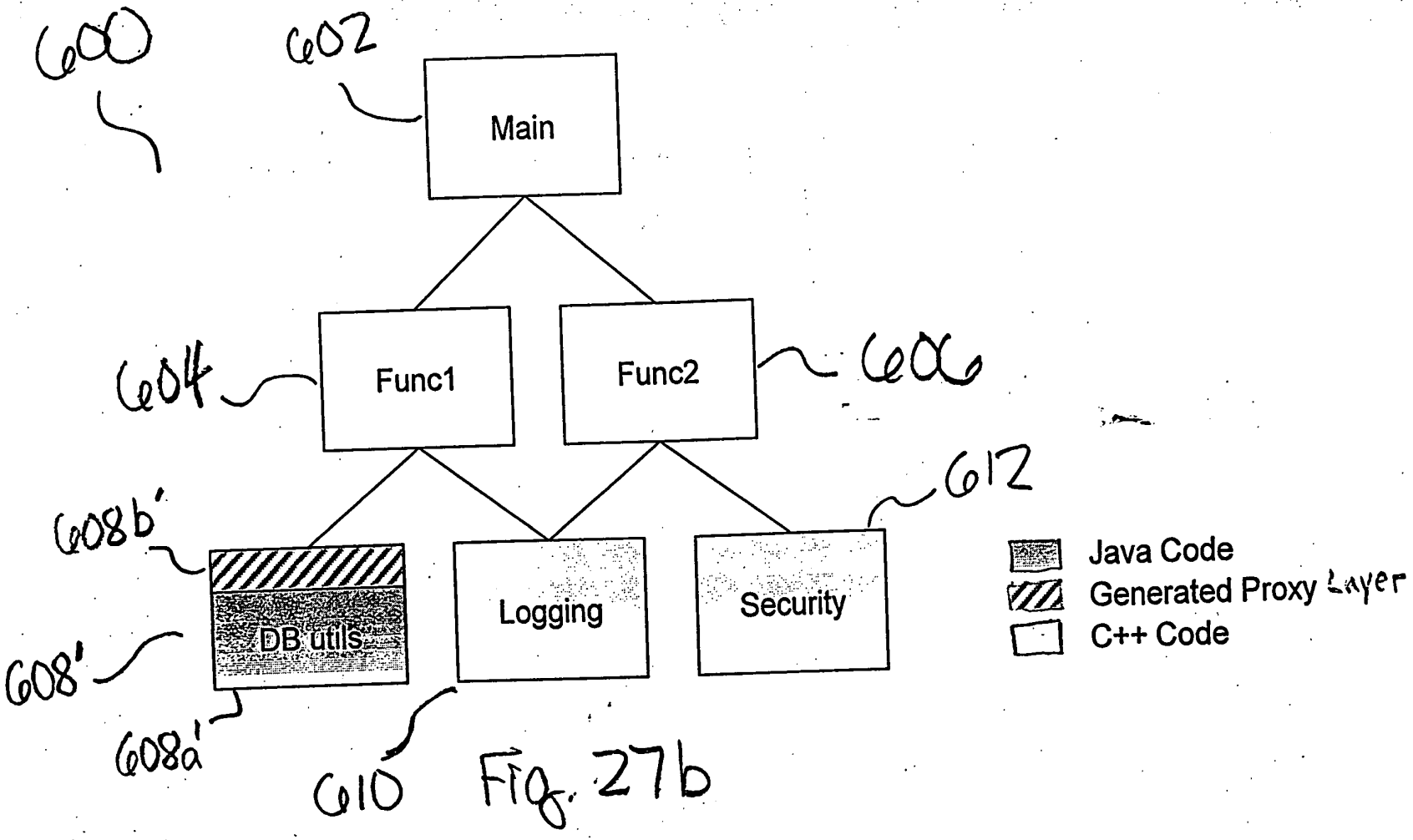

Fig. 27b

00Z1h0" 9h2T5560

Bottom-Up Port By Proxy (2nd Step)

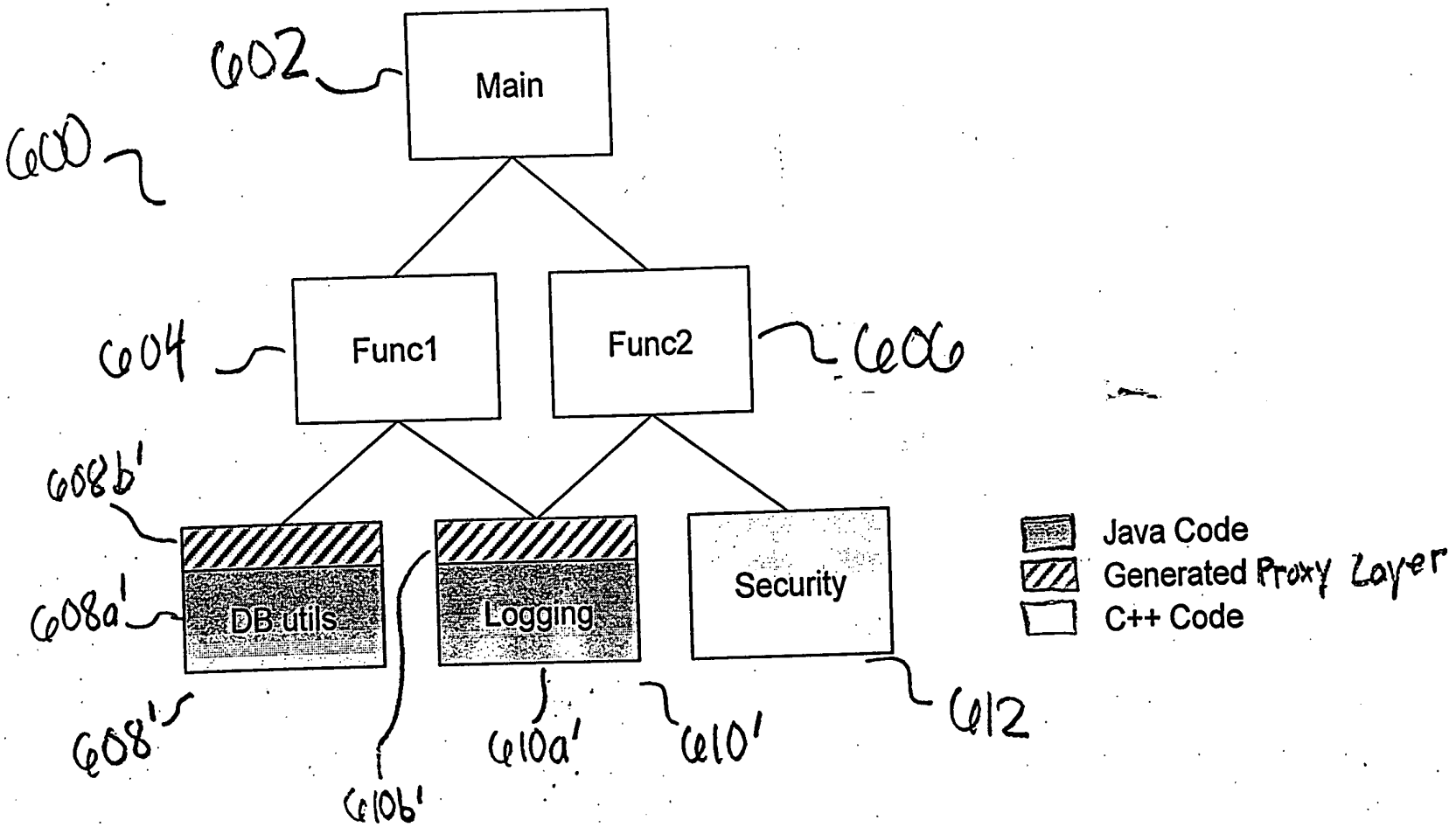

Fig. 27C

002Th0*9h2T5560

Bottom-Up Port By Proxy (3rd Step)

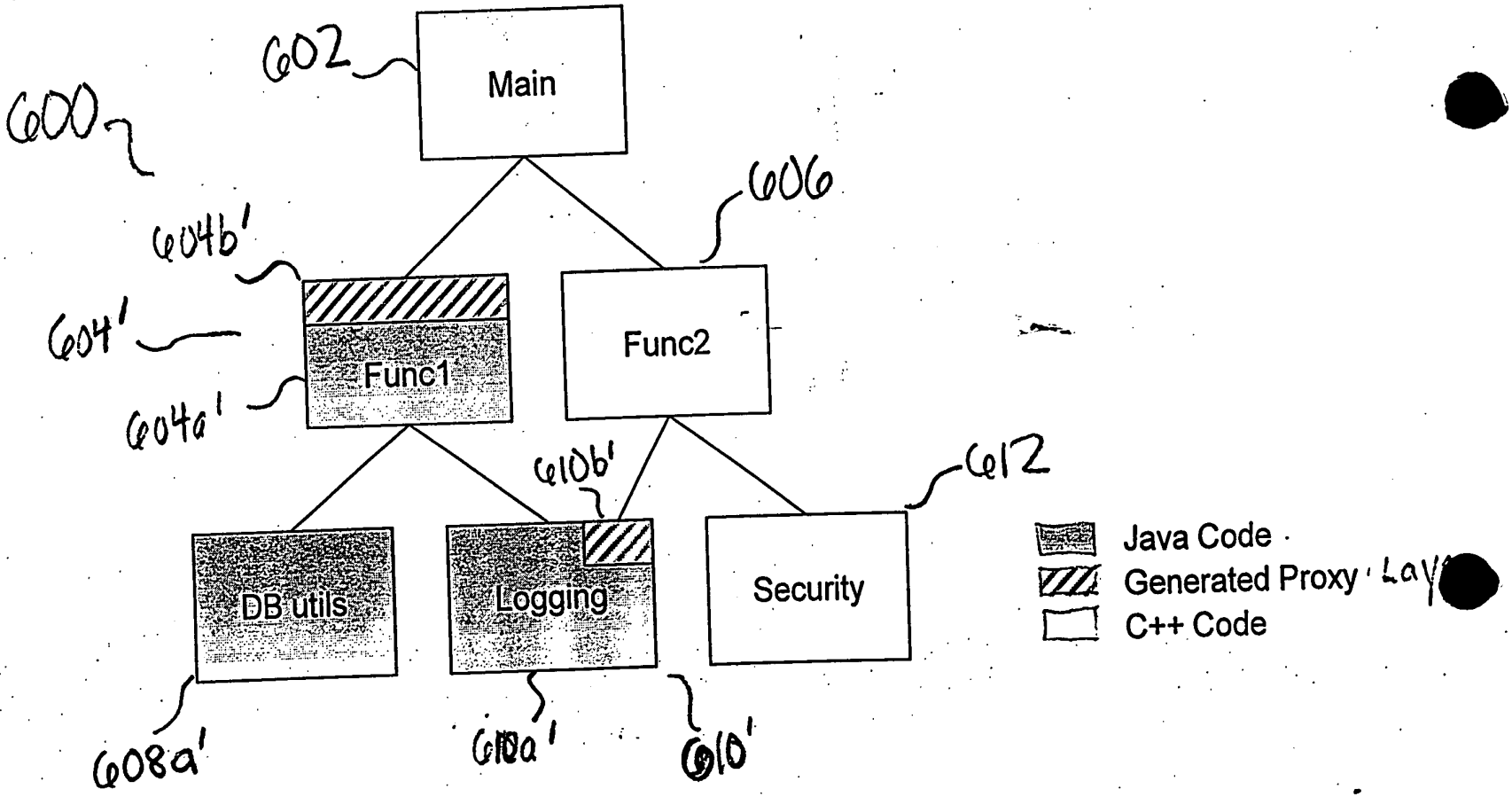

Fig. 27d

Bottom-Up Port By Proxy (4th Step)

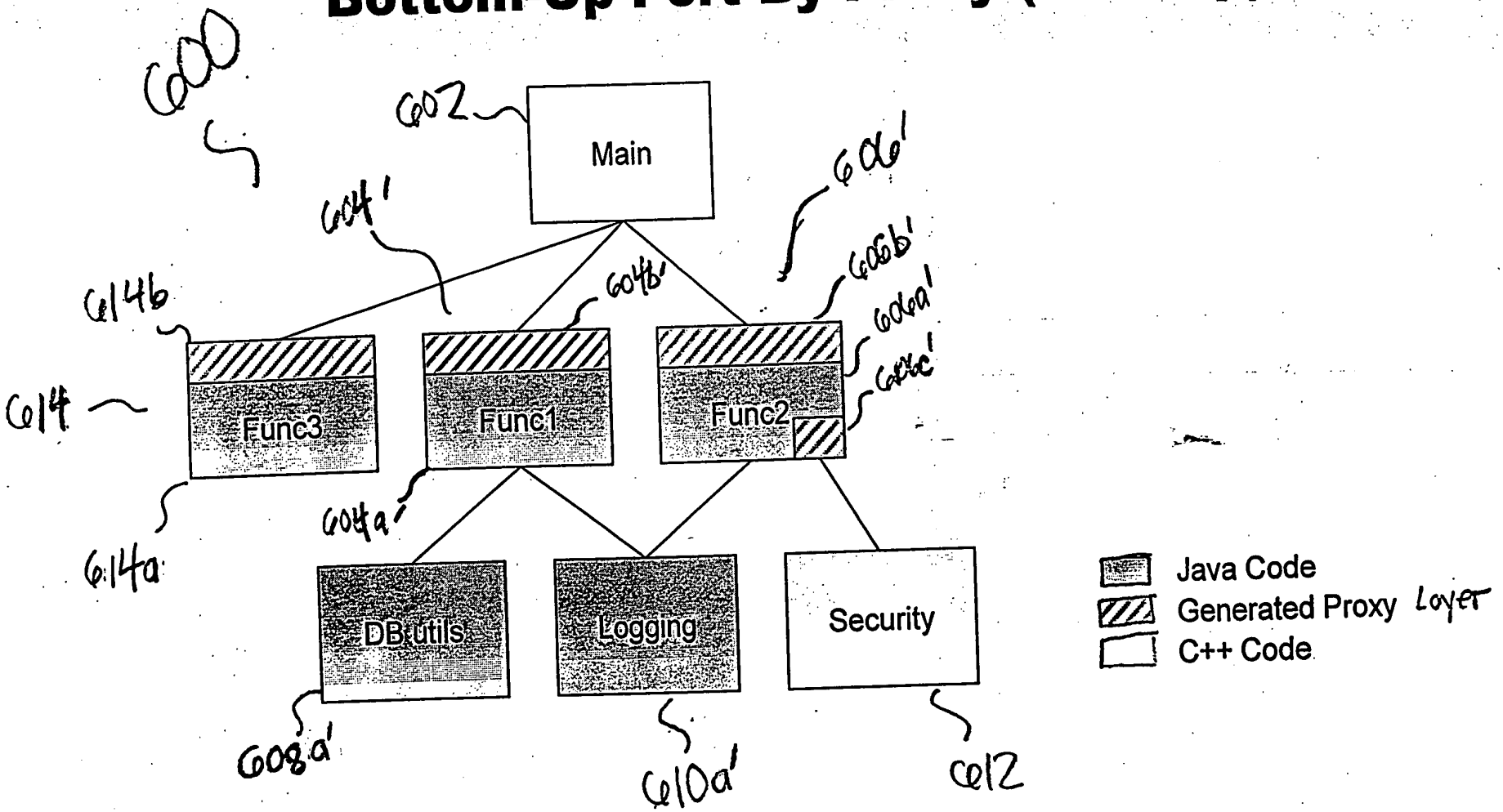

Fig. 27e

Bottom-Up Port By Proxy (5th Step)

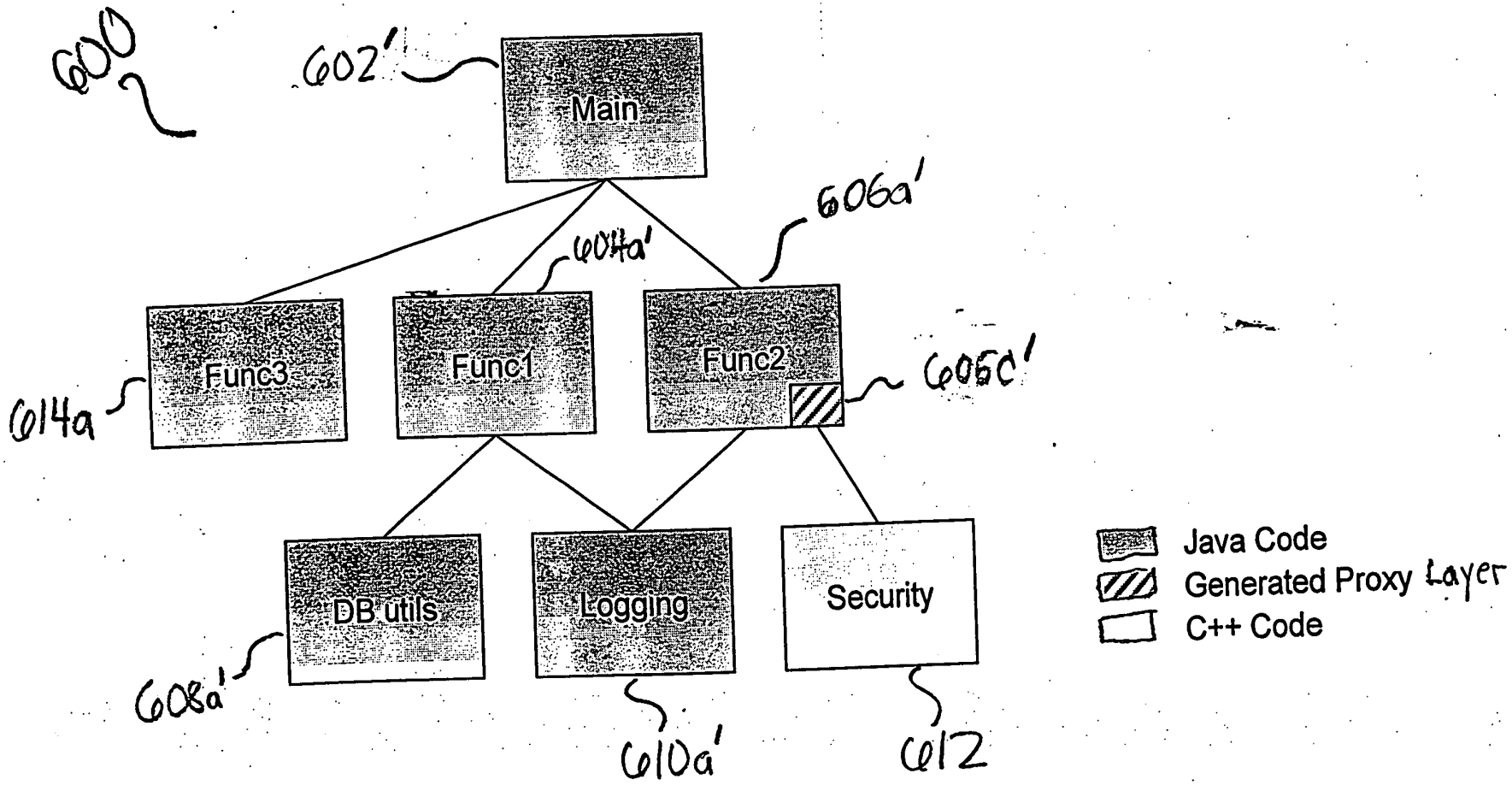

Fig. 27f